NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

THE COMPLETED MANAGEMENT INFORMATION SYSTEM FOR THE MONTEREY NAVY FLYING CLUB

by

James M. Graham

March 1987

Thesis Advisor Barry A. Frew

unclassified
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION unclassified	1b. RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.
2b DECLASSIFICATION/DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S)	5 MONITORING ORGANIZATION REPORT NUMBER(S)
6a NAME OF PERFORMING ORGANIZATION Naval Postgraduate School; 6b OFFICE SYMBOL (If applicable) 54	7a NAME OF MONITORING ORGANIZATION Naval Postgraduate School
6c. ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000	7b. ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000
8a NAME OF FUNDING/SPONSORING ORGANIZATION; 8b OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c ADDRESS (City, State, and ZIP Code)	10 SOURCE OF FUNDING NUMBERS: PROGRAM ELEMENT NO; PROJECT NO; TASK NO; WORK UNIT ACCESSION NO

11 TITLE (Include Security Classification) THE COMPLETED MANAGEMENT INFORMATION SYSTEM FOR THE MONTEREY NAVY FLYING CLUB

12 PERSONAL AUTHOR(S) Graham, James M.

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
Master's Thesis	FROM ______ TO ______	1987 March	229

16 SUPPLEMENTARY NOTATION

17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Management Information System; MIS Prototype; Limited Decision Support System; Relational Database Application

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

This thesis provides a completed Management Information System for the Monterey Navy Flying Club. The software package was designed to operate upon an IBM PC-XT or PC-AT or 100% compatible microcomputer which has 384K of main memory. Specific hardware requirements are discussed in chapter one. This software package supplies the necessary tools for the club manager to maintain all club records and generate required administrative and financial reports. Decision-making assistance is provided to the club manager and its Board of Directors by combining the system generated reports and the relational database capabilities of the R:Base 5000 language.

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT [✓ UNCLASSIFIED/UNLIMITED] ☐ SAME AS RPT ☐ DTIC USERS	21 ABSTRACT SECURITY CLASSIFICATION unclassified
22a NAME OF RESPONSIBLE INDIVIDUAL Prof. Barry A. Frew	22b TELEPHONE (Include Area Code) (408) 646-2924; 22c OFFICE SYMBOL Code 54Fw

DD FORM 1473, 84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE
unclassified

The Completed Management Information System for the Monterey Navy Flying Club

by

James M. Graham
Lieutenant Commander, United States Navy
B.S., University of Southern Mississippi, 1975

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN INFORMATION SYSTEMS

from the

NAVAL POSTGRADUATE SCHOOL
March 1987

ABSTRACT

This thesis provides a completed Management Information System for the Monterey Navy Flying Club. The software package was designed to operate upon an IBM PC-XT or PC-AT or 100% compatible microcomputer which has 384K of main memory. Specific hardware requirements are discussed in chapter one. This software package supplies the necessary tools for the club manager to maintain all club records and generate required administrative and financial reports. Decision-making assistance is provided to the club manager and its Board of Directors by combining the system generated reports and the relational database capabilities of the R:Base 5000 language.

THESIS DISCLAIMER

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

I. INTRODUCTION

A. ORGANIZATION

The Monterey Navy Flying Club is a non-profit organization located at the Monterey Peninsula Airport in Monterey, California. The club is officially sponsored by the Naval Postgraduate School (NPS) of Monterey and governed by a Board of Directors. Club administration is conducted by the club's manager who is responsible to the Board of Directors and Commanding Officer, NPS. All administrative and financial activities are conducted in accordance with FAA regulations and flight procedures, and instructions from the Department of Defense and the offices of Chief of Naval Operations (CNO).

The Monterey Navy Flying Club currently processes all administrative and financial records manually, with the exception of one accounts receivable computer program to process membership billing. Maintaining accurate club membership and financial records is a time consuming and demanding task for one individual to accomplish. In addition to the typical administrative records, the club manager must maintain detailed records on aircraft maintenance, operating costs, and membership flight status. From these separate files, data is drawn monthly and annually to prepare numerous administrative and financial reports for submission to the club's Board of Directors, the NPS Superintendent, and CNO/MNPC.

B. THESIS BACKGROUND

As one of the largest Navy sponsored flying clubs, the Monterey Navy Flying Club's membership and flight activities are numerous. To assist in effectively and efficiently managing the large volume of administrative and financial functions conducted by the club manager, an initial Management Information System (MIS) was designed and implemented [Ref. 1]. The initial MIS prototype provided limited assistance to the manager. A completed version, one capable of handling common financial transactions, maintaining membership data and flight status, recording aircraft maintenance activities and costs, and producing the numerous required reports, was desired. The system would also provide a historical database from which the club manager could recall statistical and financial data to assist in the club's strategic planning or in response to inquiries from higher authority. The system's standard reports will provide this limited decision-making support initially. The completed MIS prototype and associated relational database establish the foundation from which to implement a separate and detailed Decision Support System (DSS) application (program).

C. GOALS AND OBJECTIVES

The goals and objectives of this prototype Management Information System (MIS) are twofold: first, to provide a functioning Management Information System for the MNFC Manager's use and second, to provide an environment capable of exploring the feasibility of developing practical Decision Support Modules for the MNFC's use.

1. MNFC Management Information System

A functional MIS is intended to assist the MNFC Manager in maintaining club records, producing specific reports, significantly reducing manual duplications, operations, and time. The prototype MIS will handle current financial and administrative tasks. The following tasks/products will be performed/prepared by the system:

a. *Daily Processing*

* Membership Entry and Deletion
* Membership Charges and Payments
* Club Purchases and Payments
* Aircraft Maintenance Status

b. *General Administration*

* Prepare Member Roster
* Maintain Aircraft Inventory and Status
* Generate Manager's Monthly Report

c. *Financial*

* Aged Accounts Receivables
* Member's Monthly Statement
* Instructor's Monthly Service Statement
* Lessor's Monthly Statement
* Balance Sheet
* Income Statement

d. *Annual Report Inputs*

* Membership Breakdown
* Operating Statement

* Statement of Net Worth
* Summary of Aircraft Insurance Coverage

2. Decision Support Feasibility

The completed MIS and associated database will be used to explore the possibility of developing initial decision support modules to assist management in decision-making processes. Simple queries from the user will be employed to gain access to the various data and formulate decision support responses.

Cash transactions will not be addressed since the MNFC does not handle cash on a normal basis. Neither is inventory management of individual flight accessories nor aircraft repair parts implemented within this MIS. Any item from above can be incorporated at a later date using the R:Base 5000 software, if determined to be necessary.

D. METHODOLOGY

The initial prototype design for the management of the Monterey Navy Flying Club (MNFC) was functionally divided into two areas: (1) Manage the club's records and (2) Generate the required reports [Ref. 1].

The assessment of this original design and testing of a partial management information system for the MNFC [Ref. 1] will be discussed in Chapter II. The remaining logical design structure necessary to fully implement the MIS is presented in the data flow diagrams of Appendix A which graphically define the processes and associated data listed in Appendix C.

To satisfy the administrative requirements placed upon the manager of the MNFC [Ref. 2], the various system products are displayed in Appendix B. The User's Guide, located in Chapter III, provides the user a step-by-step guide to investigate the menu-driven system, and to experiment with the various system operations.

A sampling of the current MNFC database will be installed within the MIS prototype. The MIS functions described in detail in Appendix D will be tested. Upon a successful verification of the prototype's operations, Decision Support queries will be designed and tested. Chapter V will amplify the results of these and other efforts, and implications concerning the completion of the MNFC MIS.

E. SYSTEM REQUIREMENTS

To operate the MNFC MIS the following hardware is required:

* IBM PC-XT, PC-AT, or 100% compatible microcomputer

* DOS, VERSION 2.0 or highter

* A hard disk and one or two double-sided, double-density 5.25-inch floppy disk drives

* 384K of main memory with at least 347K available after system configuration (system processing speed is considerably enhanced if 640K of main memory is available)

* Color or monochrome monitor

* Printer, with large carriage for handling oversize printed forms, or capable of compressed printing

The Monterey Navy Flying Club's prototype Management Information System was developed utilizing the following Microrim, INC., of Bellview, Washington, software:

R:Base 5000 Database Management System

R:Base Series Extended Report Writer

F. SYSTEM OVERVIEW

The Monterey Navy Flying Club Management Information System Prototype, Version 2.0, is an interactive, user friendly, menu-driven system. The intended user's (the club Manager) man-machine interface in through a series of screen displayed menus allowing the selection of various administrative and financial functions (see Chapter III, the User's Guide).

The overall system design is based upon one relational database accessed by five functionally separate application programs. As can be seen in Figure 1.1, each application not only interfaces with the database, FLYCLUB, but also interacts with the initial calling application, FLYSTART. The individual functions performed by each application are listed in Chapter III, the User's Guide, with each module's function discussed in detail in Appendix D.

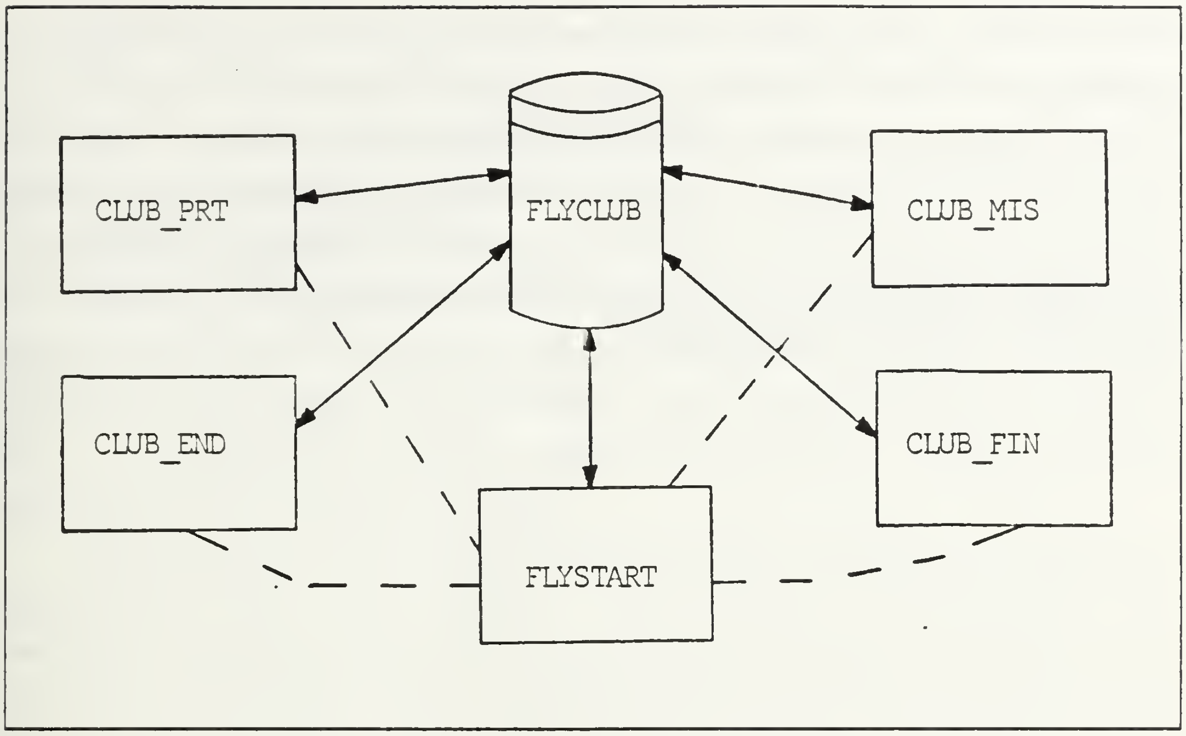


Figure 1.1 MNFC MIS Prototype System Overview.

All system products are designed to be printed on a printer capable of interfacing with an IBM compatible personal computer and either has a long carriage or a compressed print capability. The statements can be printed on preformated forms or letter head paper. All reports are considered roughs for the Manager's review and use as necessary. The system will allow the user to exit the R:BASE environment and return to the computer's operating system with the choice of the appropriate menu selection.

The R:BASE 5000 language is easily accessible for the advanced user and can be used to conduct custom searches and displays of data to cover those data inquiries not implemented in the current prototype. All database entries are stored at the completion of each function and the data history files are posted monthly.

G. DECISION SUPPORT PACKAGE

Through its standard products and data manipulation, the current MNFC MIS Prototype system provides a limited and structured decision support capability. The original system design allows for a separate application (program) to access the relational database and conduct both standard and ad hoc inquiries (question and answer sessions). This inquiry capability would give the Monterey Navy Flying Club Manager and its Board of Directors abilities such as, evaluate expenses versus revenue trends, investigate and establish membership trends, examine flight hour usage versus aircraft inventory, and perform budgetary forecasting. The actual Decision Support System (DSS) package can be designed and implemented using a combination of R:BASE basic language, R:BASE Clout, and gateway capable software to broaden the scope of inquiries and data manipulation.

II. MIS SYSTEM

A. ANALYSIS INTRODUCTION

1. Analysis and Review

In order to complete the desired objective of a final product capable of handling both the financial and administrative affairs for the Monterey Navy Flying Club (MNFC), the original software design, a partial implementation, developed by CDR. D. R. George [Ref. 1] required a second review/anaylsis to ensure that all user specifications were still current and accurate.

Both the physical and logical designs of the MNFC Management Information System (MIS) prototype were examined. These assessments were compared to the present MNFC physical operations, and resulting revisions and newly defined software requirements were obtained. Upon completion of an initial design, the author and manager of the MNFC conducted extensive system design reviews to ensure that the logical representations truly depicted the desired and current club functions.

Emphasis was placed upon the completion of the financial and annual reporting functions of the MNFC MIS. To enhance user friendliness, revisions in areas such as menu selection, entry formats and output formats to the original design were conducted. The final expectation for the prototype has not been altered.

2. The Analyst and User Interaction and Exchanges

As in any software analysis and design, perhaps the most critical problem facing the analyst is ensuring the user's requirements and desires are known. Quite too often neither the user nor the system analyst knows what is really required or desirable [Ref. 3].

Prototype design fosters greater user interaction in the .software design and helps prevent the development and delivery of software systems that do not satisfy the user's real needs.

During the fourteen discussions and working sessions held by the author and MNFC manager, existing man-machine interfaces, software products, and operational procedures were reviewed. Over the course of these exchanges, the originally designed menus were altered to better meet the user's preception of the software function to logical operation. The resulting menus are located in Chapter III. The existing product outputs of the Management Information System were modified to meet both new and revised requirements. Current product examples can be seen in Appendix B.

These exchanges provided an excellent communication medium for the author to present output alternatives to the MNFC Manager. This gave the manager the opportunity to see the proposed system products and reassured him that his needs and desires were truly being understood and met. This demonstrated the author's sincere effort to accommodate the user's wishes; thereby, creating a better environment for further communications concerning the development of the Management Information System prototype.

B. SECOND ANALYSIS RESULTS

1. System Design Requirements

a. Areas to Redesign

Upon the completion of the author's initial analysis and review, redesign of the following areas in the original design were evident:

(1) *Logical Design.*

* General DFD revisions

* Financial/Administrative interfaces

* Additions and Revision of tables

(2) *Man-Machine Interface.*

* System Start-up procedures

* Menu title changes

* Entry of reoccurring events

* System Data Saving

* System Recovery

(3) *Product Outputs.*

* Deletion of the Dispatcher Report

* Redesign of the Member Roster

* Revise the Lessor Statement

* Enchance the Member Statement

b. Areas Remaining to be Designed

To complete the Monterey Navy Flying Club (MNFC) Management Information System prototype, the following areas required design and development:

(1) *Logical Design.*

* Financial operations

* Product Output Generation

* Administrative/Finance interface

* Aircraft Maintenance and inventory

* MNFC Purchases and Payments

(2) *MIS Products.*

* Balance Sheet

* Income Statement

* Operating Statements

* Statement of Net Worth

* Summary of Insurance coverage

* Membership Breakdown

* Aged Accounts Receivable Report

* Aircraft Inventory & Status Report

* Manager's Monthly Report

2. System Design Constraints

One of the key objectives in the design was to develop a system that was user friendly. Normal operations are to be executed in a fast and efficient manner, allowing the user to spend the least amount of time at the keyboard. This required some data redundancy within the various database table designs. The execution of some operations; i.e., the posting of reoccurring events, are accomplished at the printing of the system product, and causes some time delay, but does not require the presence of the user.

The system itself requires the use of a hard disk to reduce storage and overall system execution time requirements. Therefore, hard disk utilization provides adequate disk growth space and eliminates the need for multiple floppy disk changes; thus, improving user friendliness and ease of operation.

C. OVERVIEW OF THE COMPLETED DESIGN

1. Resulting Logical Design

The Data Flow Diagrams (DFD) contained in Appendix A represent the logical design necessary to implement the Monterey Navy Flying Club (MNFC) MIS prototype software for current and expected physical functions of the club. As seen in

Figure A.1, the MNFC's interaction is restricted to three major outside forces: (1) the club manager/staff, (2) club members, and (3) service vendors. Upon further reveiw of the club's operations one notices in Figure A.2 that the overall system is divided into two major functional areas: (1) maintaining club administrative and financial records and (2) generating required reports.

Figure A.3 through Figure A.25 graphically present the individual club functions necessary to maintain the daily operations of the club and to produce the system products exhibited in Appendix B. Future maintenance personnel will be able to use the DFD's to ensure the logical operations are consistent with the physical operations, as changes to club routines and procedures occur.

2. Physical Revisions and Design

The Monterey Navy Flying Club (MNFC) does not handle, on a regular basis, any cash transactions. All members and vendors rendor or receive payment by check. Statements are issued monthly to members, instructors, and lessors. The original prototype software design was altered to eliminate all cash transactions.

The elimination of cash operations did not abolish the requirements for recording the club's financial interactions with the public. Club purchases and payments are now incorporated within the newly designed financial operations. Inventory of flight supplies, fuel, etc. can now be monitored. Revenues from sales are posted with expenses, so net income (loss) can be calculated.

Lessors were once given credit for flight hours flown on their aircraft. Due to a recent policy change, lessors are now charged for aircraft rental just as a regular member (a/c rental cost are no longer deducted from the amount owed lessor). The Lessor Statement was changed to reflect this change in club policy. Another policy change prompted the removal of the Dispatcher's Report, since the data was no longer retained nor required.

In order to provide a better understanding of actual flight hours flown by individual membership categories, such as Naval Officer, Naval Enlisted, Army Officer, revisions were necessary to the Flight Hours and Flight History tables to accommodate the addition data fields. This revision not only provided management a broader prospective of flight usage by members, but also fulfills the Chief of Naval Operations (CNO) annual report requirement for flight hours flown by membership types/categories.

To facililate posting special dues and charges to the individual accounts, the revised MNFC MIS prototype can now handle "reoccurring events". The MNFC staff/manager can code each account so that during the end-of-the-month transactions a predetermined membership due, fee, or any other special charge that reoccurrs on a regular basis, can be posted without the user's assistance.

3. Verification of User Requirements

The logical design of the Monterey Navy Flying Club (MNFC) Management Information System (MIS) prototype was discussed and reviewed often with the MNFC manager. After each revision the logical Data Flow Diagrams (DFD) we re-examined and compared to the actual club operations to ensure functional accuracy.

The MNFC manager, the major user of the prototype system, was presented example products of the MIS. These products were designed from the user's specifications previously obtained, and any desired changes identified by the users. The revised products were once again reviewed with the final copies approved prior to coding.

This prototyping design technique should prevent the user from seeing an unfamiliar item, both in daily operation of the system and in its products. Plus, this method of design should reduce the amount of corrective maintenance on function procedures and products after system delivery.

Prior to delivery of the MNFC MIS software package, the prototype will be subjected to an operational test utilizing a sample of the current MNFC database. The results will be compared for accuracy with those produced under the present manual system.

III. USER'S GUIDE

A. INTRODUCTION

The following user's guide is written under the assumption that its intended user, the Manager of the Monterey Navy Flying Club, will be utilizing an IBM XT or a 100% IBM compatible computer with a 5.25-inch floppy disk drive, a minimum of 347K internal memory after system configuration, a 10MG Byte hard disk, a graphics printer with either a long carriage or a compressed print capability, and using the MS/DOS operating system. If a different hardware configuration is used, the user is directed to the owner's manual to see how to load and operate the MNFC MIS Prototype and its associated database, FLYCLUB.

The MNFC MIS Prototype was designed to assist the Monterey Navy Flying Club Manager by maintaining the club's administrative and financial records, preparing monthly billing, printing both monthly reports and annual report inputs, and providing a current and historical database from which data inquiries could be made. Data drawn from this historical database would help the Club Manager and the Board of Directors in decision-making processes. The following topics are discussed in this user's guide to aid the user with system operations:

- System Start-up Procedures
- Database Initialization
- MNFC MIS Operations
- System Back-up and Database Repacking Procedures
- Exiting the MNFC Prototype
- Maintenance

B. SYSTEM START-UP PROCEDURES

1. Machine Configuration Requirement

First the user must ensure that his machine environment is properly initialized with a CONFIG.SYS file which has the following minimum settings:

- Buffers = 5
- Files = 20

If the CONFIG.SYS file was not resident during boot initialization or did not comply with the stated requirements above, the user will need to build or modify the system CONFIG.SYS file accordingly. The user should refer to the MS/DOS Manual for explanatory steps necessary to write or alter a CONFIG.SYS file. Once the correct CONFIG.SYS file is developed and stored, the computer should be booted or rebooted by hitting the following three keys at the same time: <CTRL> <ALT> <DEL>. Unless the system has a clock installed, the computer will ask for the current date and time. The user must enter the correct date and time in order for the MIS Applications to post the database table entries correctly since system time and date are used by numerous applicational modules for financial posting. The user will hit the <ENTER> key after each date and time entry.

2. Initial Program Loading

Now that the computer has been rebooted and the correct date and time have been entered, the screen should have the 'C' drive prompt displayed, which looks like this: "C:>". Take from storage the following seven floppy disks: the three R:BASE 5000 System disks, the two Master MNFC MIS Prototype disks, and the two FLYCLUB database disks. Place the R:BASE 5000 System Disk #1 into disk drive 'A'. Now change the default drive to the 'A' drive by entering the command ":A" After hitting the <ENTER> key, the 'A' prompt (A:>) should now be present on the active screen line noted by the blinking cursor. Copy the disk material onto the hard disk with the command:

```
COPY  *.*  C:  <ENTER>
```

When the "A:>" prompt reappears with the cursor blinking beside it to the right you are ready to load the second system disk onto the hard disk by repeating the above procedure. Repeat this process until all three R:BASE 5000 System Disks, the MNFC MIS Prototype System Disks, and the FLYCLUB database disks have been copied onto the computer's hard disk. Upon completion of copying the disks the user's computer now will have all necessary software to execute the MNFC MIS Prototype System. The MIS Prototype Applications and the FLYCLUB database structure will also be present on the hard disk.

The user could also place all MIS Prototype associated software under a separate sub-directory title on the hard disk. The user should follow the instructions listed within the MS/DOS Manual to create a sub-directory for the MIS system. Once the sub-directory is built, enter the sub-directory and then copy all seven system disks as previously discussed onto the hard disk. All MIS Prototype operations should be executed from within the newly designated sub-directory.

Now the user needs to change the default drive to that of the hard disk by entering at the "A:>" prompt: "C:" and hit the <ENTER> key.

3. Other Initializing Preparations

With the computer system ready to run the MNFC MIS Prototype, the user needs to have a minimum of ten preformatted 5.25-inch floppy disks to be used when called by the programs to back-up the FLYCLUB database. THE USER WILL NOT BE ABLE TO FORMAT ANY DISKS WHILE EXECUTING THE PROTOTYPE AND COULD LOSE ALL DATA WITH SYSTEM SHUTDOWN. System shutdown will be the only method to exit the prototype if the user is unable to complete execution of a system back-up/repack routine.

To format the disks, just insert a new disk into disk drive 'A' and enter the command:

```
FORMAT A:   <ENTER>.
```

Follow the directions displayed on the screen until all ten disks have been properly formatted. Store these disks separately from your unformatted disks in preparation for their use by either the "Post Monthly Journal Entries" or "Back Up and Pack the Database" program modules.

Some operations require the use of a printer. The correct predesigned printer forms (e.g. Monthly Member Statements) or the required printer paper should be loaded and the printer energized before execution of the MNFC MIS Prototype.

C. DATABASE INITIALIZATION

1. Initialization Requirements

During program execution, the MNFC MIS Prototype receives data inputs via various screen display entry forms. However, before the menu driven functions of the prototype can be used, certain tables of the FLYCLUB database must be initialized/loaded; such as, initial information on each aircraft, current account balances, etc. The following tables must be initialized prior to system operations:

- BAL_SHET -- Current balances for each account
- EARNINGS -- Current balances of income and expense accounts
- A/C_REC -- Individual aircraft data
- A/C_HRS -- Current hob readings per aircraft
- CAP_ASET -- Current Capital Asset status
- ACCT_REC -- Current Accounts Receivable balances
- ACCT_PAY -- Current creditors' data and account balances

- MBR_SUM -- Current flight hours flown by each member category

For ease and speed, membership records should be initialized using option (1) under the Membership Transactions Menu (see Figure 3.2, on page 28).

2. Initialization Procedures

In order to initialize the FLYCLUB database, the user must first disenable the automatic exec feature of the MNFC MIS Prototype. At the C:> prompt, after the system has been turned on and the current date and time entered, the user should execute the command:

```
RENAME   RBASE.DAT   RBASE.TXT
```

The RBASE.DAT command file calls the MNFC MIS Prototype FLYSTART Application program and starts the menu driven system. With this file disabled the user can now use the underlying R:BASE software to load the FLYCLUB database.

Enter the command: R:BASE and press <ENTER>. The user will see a large "R" appear, followed by the R:BASE Menu. Select option (1) and press the <ENTER> key. At this time the R:> prompt will be present. Type OPEN FLYCLUB and press <ENTER>. For each table listed in paragraph C.1 to be initially loaded, execute the command:

```
LOAD <table name> WITH PROMPTS  <ENTER>
```

Then provide the data requested. Once the table has been loaded, press the <ESC> key, which will return the user to the R:> prompt. Repeat the process until all necessary tables have been initialized. Once done, type CLOSE and press <ENTER>.

Now that the database has been entered, it is best to save the data in case of a system failure. Place a formated disk in disk drive 'A' and copy FLYCLUB1.RBS and FLYCLUB3.RBS. Then remove the disk and insert another to copy FLYCLUB2.RBS. The command format for copying is:

```
COPY FLYCLUB<number>.RBS A:
```

Remove the last disk, label and store both disks in a safe place.

The last item to set is the R:BASE date function. The date can be entered and displayed in one of several formats: day-month-year, day-year-month, month-day-year, month-year-day, year-month-day, or year-day-month. The day is expressed as two digits while the month can be expressed as either two digits or three letters. The year can be displayed as either two or four digits. Currently, the MNFC MIS Prototype is in the R:BASE default setting of MM/DD/YY. All dates must be entered in that format to be accepted.

If the system default format is acceptable, skip to subsection 4.C.3. To set the date format to display the date such as JAN 26,1987, enter the command:

```
SET DATE MMM-DD-YYYY  <ENTER>.
```

At this time, enter the date format desired. [Ref. 4: pp. 82-83]

3. Finalize Initialization

Now that the database is loaded, saved, and the date format is chosen, the user must exit R:BASE and reenable the auto executive command file. To accomplish these tasks enter the following commands at the noted prompt:

```
R:> CLOSE  <ENTER>
R:> EXIT  <ENTER>
C:> RENAME  RBASE.TXT  RBASE.DAT  <ENTER>
```

D. MNFC MIS PROTOTYPE OPERATIONS

1. Menu Driven System

The man-machine interface for this Management Information System prototype is a series of menus. Starting with the root menu, The MNFC MAIN MENU, see Figure 3.1, the user branches to the desired system function/operation. Each selected menu allows the user to return to the previously called menu, eventually returning to the MNFC MAIN MENU. All menu selections are grouped functionally to provide a more efficient working environment for the user (e.g. all member charge transactions can be accomplished in one session without returning to the main menu for each entry).

Examples of the remaining menu branches and their applicable sub-branches are displayed in Figure 3.2 through Figure 3.14. Detailed functional descriptions of each selectable application (program) module can be found in Appendix D.

To start the MNFC MIS Prototype system and to get the main menu to appear, the user types at the C:> prompt the command:

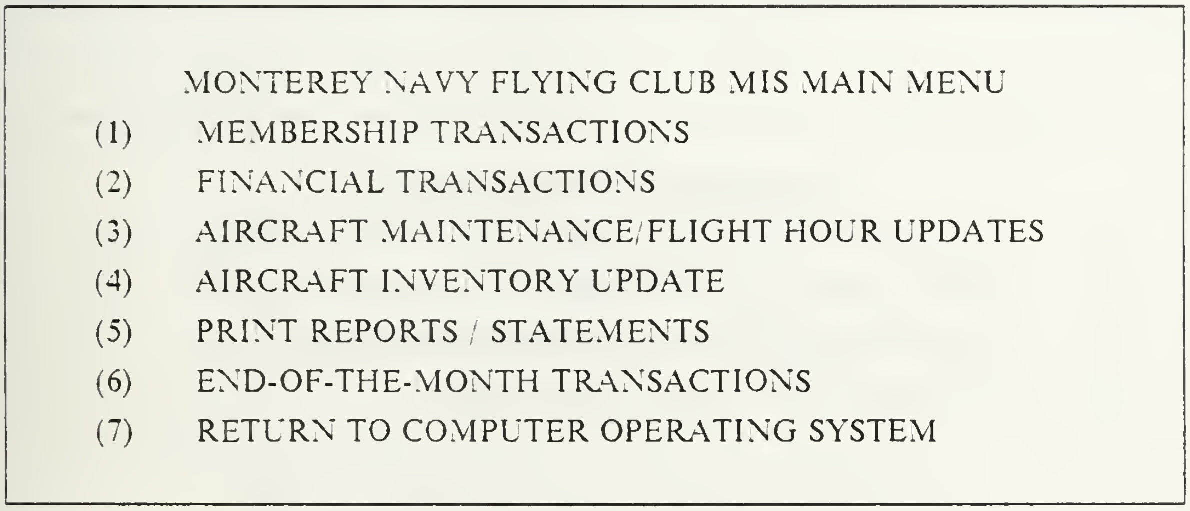
```
RBASE  <ENTER>
```

Once the MNFC Main Menu appears, see Figure 3.1, the user is free to choose any one of the operations by moving the highlighted cursor up or down with the proper arrow key, then hit the <ENTER> key. The user will either be presented another menu from which to choose a specific function or begin execution of the designated function. Short descriptions of each menu selection follows in section 3.D.2.

2. **MNFC Selections**

a. MNFC Main Menu (Figure 3.1)

Selecting option (1), MEMBERSHIP TRANSACTIONS, displays the MEMBERSHIP TRANSACTIONS Menu (Figure 3.2). Option (2) calls the FINANCIAL TRANSACTIONS Menu (Figure 3.3). AIRCRAFT MAINTENANCE/FLIGHT HOUR UPDATES are accomplished by selecting option (3). The AIRCRAFT INVENTORY UPDATE is handled when option (4) is chosen. Option (5) calls the PRINT REPORTS/STATEMENTS Menu (Figure 3.9). The END-OF-THE-MONTH TRANSACTIONS Menu (Figure 3.13) is display upon selection of option (6). With the selection of option (7) the user exits the prototype and returns to the computer operation system.



```
        MONTEREY NAVY FLYING CLUB MIS MAIN MENU
(1)      MEMBERSHIP TRANSACTIONS
(2)      FINANCIAL TRANSACTIONS
(3)      AIRCRAFT MAINTENANCE/FLIGHT HOUR UPDATES
(4)      AIRCRAFT INVENTORY UPDATE
(5)      PRINT REPORTS / STATEMENTS
(6)      END-OF-THE-MONTH TRANSACTIONS
(7)      RETURN TO COMPUTER OPERATING SYSTEM
```

Figure 3.1 MNFC Main Menu.

b. Membership Transactions (Figure 3.2)

The user is provided three choices in this menu; Enter a New Member Record, Edit/Delete an Existing Member Record, and Return to the MNFC Main Menu. The first two choices use interactive screen entry forms to request new member data or allow the edition or deletion of current member data. See section 3.3 and section 3.4 on editing and entering procedures.

c. Financial Transactions (Figure 3.3)

Option (1) displays the MANAGER'S UPDATE ENTRY Menu (Figure 3.4). New financial transactions are entered via option (2) which displays the ENTER

```
                    MEMBERSHIP TRANSACTIONS
    (1)     ENTER A NEW MEMBER RECORD
    (2)     EDIT / DELETE AN EXISTING MEMBER RECORD
    (3)     RETURN TO THE MNFC MAIN MENU
```

Figure 3.2 Membership Transactions (Branch Menu).

A NEW FINANCIAL TRANSACTION Menu (Figure 3.7). All financial corrections and deletions are accomplished from the EDIT/DELETE AN EXISTING TRANSACTION Menu option (3). The fourth option returns the user to the MNFC Main Menu.

```
                    FINANCIAL TRANSACTIONS
    (1)     MANAGER'S UPDATE ENTRY
    (2)     ENTER A NEW TRANSACTION
    (3)     EDIT / DELETE AN EXISTING TRANSACTION
    (4)     RETURN TO THE MNFC MAIN MENU
```

Figure 3.3 Financial Transactions (Branch Menu).

d. Manager's Update Entry Menu (Figure 3.4)

The manager can correct any financial account either on the Balance Sheet or the Income Statement depending upon which option is chosen. Option (1) displays the Balance Sheet Adjustment Menu (Figure 3.5), while option (2) gives the user access to the Income Statement Adjustment Menu (Figure 3.6). Option (3) returns the user to the main Financial Transactions Menu.

e. Balance Sheet Adjustment (Figure 3.5)

All asset accounts carried upon the club's Balance Sheet can be edited, corrected, and/or loaded by selecting option (1). The club's liability accounts can also be adjusted, as with the asset accounts, by selecting option (2). The user must provide

```
                 MANAGER'S UPDATE ENTRY
(1)   BALANCE SHEET ADJUSTMENT
(2)   INCOME STATEMENT ADJUSTMENT
(3)   RETURN TO FINANCIAL TRANSACTION MENU
```

Figure 3.4 A Manager's Update Entry (Sub-Branch).

the correct account number, the amount of adjustment, and whether the adjustment is an increase or decrease in the account's balance. Option (3) returns the user to the Manager's Update Entry Menu allowing adjustment entries to the income and expense accounts or exiting to the main Financial Transaction Menu.

```
                BALANCE SHEET ADJUSTMENT
(1)   ASSET ACCOUNT ENTRY
(2)   LIABILITIES ACCOUNT ENTRY
(3)   RETURN TO MANAGER'S UPDATE ENTRY MENU
```

Figure 3.5 Balance Sheet Adjustment (Sub-Branch).

f. Income Statement Adjustment (Figure 3.6)

Any income or expense account on the club's Income Statement can be edited, corrected, and loaded under this menu. Revenue/Sale accounts are adjusted by selecting option (1) and expense accounts with the selection of option (2). The user must provide the correct account number, the amount of adjustment, and whether the adjustment is an increase or decrease to the account's balance. Option (3) returns the user to the Manager's Update Entry Menu.

g. Enter A New Financial Transaction (Figure 3.7)

The user is able to select one of five typical financial transactions from this menu. The last option returns him to the main Financial Transactions Menu. Once again interactive entry forms are used to receive data from the user and then load the

```
            INCOME STATEMENT ADJUSTMENT
(1)    REVENUE/SALE ACCOUNT ENTRY
(2)    EXPENSE ACCOUNT ENTRY
(3)    RETURN TO MANAGER'S UPDATE ENTRY MENU
```

Figure 3.6 Income Statement Adjustment (Sub-Branch).

database. To enter one or more member charge sheets, the user selects option (1). All payments upon their accounts by members can be entered with option (2). With both options the user can make multiple entries and exit at will. Credit purchases are entered with option (3) and all cash purchases are covered with option (4). Club payments toward accounts payable accounts are handled by selecting option (5). Option (6) returns the user to the main Financial Transactions Menu.

```
          ENTER A NEW FINANCIAL TRANSACTION
(1)    ENTER A MEMBER'S CHARGES
(2)    ENTER A MEMBER'S PAYMENTS
(3)    POST AN ACCOUNTS PAYABLE PURCHASE/INTEREST
(4)    POST CASH PURCHASE OF SERVICES/GOODS
(5)    POST PAYMENT TO AN ACCOUNTS PAYABLE ACCOUNT
(6)    RETURN TO FINANCIAL TRANSACTIONS MENU
```

Figure 3.7 Enter a New Financial Transaction (Sub-Branch).

h. Edit/Delete An Existing Transaction (Figure 3.8)

Corrections to charges and payments whether by a member or the club can be accomplished while in this menu. The system will ask if the error has been discovered prior to monthly posting or afterwards. Each case is handled differently; therefore, the user must be aware of when the error was found. The account data and corrective amount are required before entering the selection. Member charge and payment corrections are accomplished by options (1) and (2) respectively. The club charge corrections are handled with option (3) while payments are corrected by option (4). Option (5) returns the user to the main Financial Transactions Menu.

```
          EDIT / DELETE AN EXISTING TRANSACTION
(1)   MEMBER CHARGE TRANSACTION
(2)   MEMBER PAYMENT TRANSACTION
(3)   ACCOUNTS PAYABLE PURCHASE
(4)   PAYMENT TO AN ACCOUNTS PAYABLE ACCOUNT
(5)   RETURN TO FINANCIAL TRANSACTIONS MENU
```

Figure 3.8 Edit/Delete an Existing Financial Transaction (Sub-Branch).

i. Print Reports/Statements (Figure 3.9)

This menu is the main printing menu which calls the appropriate submenu to handle the printing requirements for Administrative Reports, Financial Reports, and Annual Report Inputs. Option (1) calls the Administrative Reports Menu (Figure 3.10). The Financial Reports' menu (Figure 3.11) is selected by option (2). And the printing of inputs to the Annual Report is done via the Print Annual Report Inputs menu (Figure 3.12) with option (3). As in other menus, the last option, number (4), returns the user to the MNFC Main Menu. The system printer should be energized and ready for operation prior to selecting a print routine.

```
              PRINT REPORTS / STATEMENTS
(1)   PRINT ADMINISTATIVE REPORTS
(2)   PRINT FINANCIAL REPORTS
(3)   PRINT ANNUAL REPORT INPUTS
(4)   RETURN TO THE MNFC MAIN MENU
```

Figure 3.9 Print Reports / Statements (Branch Menu).

j. Print Administrative Reports (Figure 3.10)

Three commonly used reports are produced under this menu. Option (1) searches the FLYCLUB database and prints the current Membership Roster (Figure B.1). The Manager's Monthly Report (Figure B.3) is produced in the rough for his review and approval by selecting option (2). A current status report of all club owned or leased aircraft, the Aircraft Inventory and Status Report (Figure B.2), is obtained by using option (3). Access to the main Print Report/Statement Menu is regained by selecting option (4).

```
                PRINT ADMINISTRATIVE REPORTS
(1)     PRINT MEMBER ROSTER
(2)     PRINT MANAGER'S MONTHLY REPORT
(3)     PRINT A/C INVENTORY & STATUS REPORT
(4)     RETURN TO PRINT REPORTS/STATEMENTS MENU
```

Figure 3.10 Print Administrative Reports (Sub-Branch).

k. Print Financial Reports (Figure 3.11)

The user can have one of three useful financial documents produced from within this menu; Aged Accounts Receivable Report, a Balance Sheet, or an Income Statement. The account balances in each document are as accurate as the last monthly postings. Option (1) gives the manager the Aged Accounts Receivable Report (Figure B.4). The Balance Sheet (Figure B.9) is produced by selecting option (2) and the Income Statement (Figure B.8) with option (3). Control is returned to the Print Report/Statement Menu with option (4).

l. Print Annual Report Inputs (Figure 3.12)

Inputs (rough drafts) to the required annual reports (Membership Breakdown, Annual Operating Statements, and Summary of Insurance) can be obtained from these menu selections. The breakdown of flight hours flown by each membership category and FAA-certification is produced at year's end by selecting option (1). Data for this report is posted monthly with the MBR_SUM table and can be printed only once per year as the Membership Breakdown Summary Report (Figure

```
                PRINT FINANCIAL REPORTS
(1)     PRINT AGED ACCOUNTS RECEIVABLE
(2)     PRINT BALANCE SHEET
(3)     PRINT INCOME STATEMENT
(4)     RETURN TO PRINT REPORTS/STATEMENTS MENU
```

Figure 3.11 Print Financial Reports (Sub-Branch).

B.10). Option (2) produces five input reports which make-up the annual operating statements required by OPNAV (see Figure B.11a through Figure B.13). Option (3) prints a summary of all listed aircraft insurance information (see Figure B.14). Control is returned to the Print Report/Statements Menu with option (4).

```
              PRINT ANNUAL REPORT INPUTS
(1)     PRINT MEMBERSHIP BREAKDOWN
(2)     PRINT ANNUAL OPERATING STATEMENTS
(3)     PRINT SUMMARY OF INSURANCE
(4)     RETURN TO PRINT REPORTS/STATEMENTS MENU
```

Figure 3.12 Print Annual Report Inputs (Sub-Branch).

m. End-of-the-Month Transactions (Figure 3.13)

From within this menu the user posts all reoccurring charges, prepares all bills and statements, closes the monthly journal, posts all accounts, makes a backup copy of the FLYCLUB database, and repacks the database. The BACKUP AND PACK THE DATABASE option should be performed weekly. Posting the reoccurring charges must be accomplished before any billing. Following the billing, the remaining journal entries can be posted. Except for the weekly backup and repack the user should execute each option in order. Option (1) is chosen to post all reoccurring charges, such as membership dues. Option (2) displays the menu to handle all monthly billing and statement production (Figure 3.14). Option (3) completes the end-of-the-month postings. Database backup and repack are done under option (4). Control is returned to the MNFC Main Menu by selecting option (5).

```
              END-OF-THE-MONTH TRANSACTIONS
   (1)    POST REOCCURRING CHARGES
   (2)    POST AND PREPARE MONTHLY BILLING
   (3)    POST MONTHLY JOURNAL ENTRIES
   (4)    BACK UP AND PACK THE DATABASE
   (5)    RETURN TO THE MNFC MAIN MENU
```

Figure 3.13 End-of-the-Month Transactions (Branch Menu).

n. Post and Prepare Monthly Billing (Figure 3.14)

The posting and preparation of monthly billing and statements was separated into three areas to allow the user to complete one billing/statement process then handle other transactions as necessary. Additionally, the printer forms for each operation are different and this separation allows for printer set up. Option (1) posts all member charges and payments then produces a monthly bill (Figure B.6) for each active member. Option (2) computes all instructor flight time and produces an instructor's statement (Figure B.5). The lessor statements (Figure B.7) are computed and printed by selecting option (3). The user is returned to the previous menu with option (4).

3. Editing/Deleting Screens

When executing the menu selections for editing and deleting records, the user will have displayed at the top of his screen, above the predesigned edit/delete form, a horizontal menu:

---Skip---Edit---Change---Add---Reset---Delete---Quit---

The user's cursor will normally be highlighting the text within the predesigned edit/delete form awaiting editing. The user will move the highlighted cursor with the <TAB> and <SHIFT-TAB> keys, and make the necessary corrections in the usual manner, using the arrows, <INS>, and <DEL> keys. Any special instructions peculiar to that edit/delete form will be displayed at the bottom of the screen.

```
                POST AND PREPARE MONTHLY BILLING
   (1)     POST AND PRINT MEMBER STATEMENTS
   (2)     POST AND PRINT INSTRUCTOR STATEMENTS
   (3)     POST AND PRINT LESSOR STATEMENTS
   (4)     RETURN TO THE END-OF-THE-MONTH MAIN MENU
```

Figure 3.14 Post and Prepare Monthly Billing (Branch Menu).

Once the change has been made, press the <ESC> key unless other instructions are provided. The cursor will be moved to the top horizontal menu. Select "Change" to send the modification to the table. Then select "Quit" to save all changes made. For deletion, select "Delete" to remove the displayed data from the associated table. The system will ask for confirmation of each deletion request. After confirming the deletion, the data will be removed and then the user must select "Quit". The remaining options within the horizontal menu are rather self explanatory. [Ref. 4: pp. 122-125]

4. Screen Enter Forms

When entering data into a predesigned screen entry form, the user can type data into the current highlighted cursor area. Normal procedures allow for the use of the <ENTER> key to move the highlighted cursor to the next data element to be completed. However, any special instructions for a specific screen entry form will be displayed at the bottom of the screen. These instructions supersede the norm and must be followed. The user can skip data elements by using the <TAB> key. The arrow, <INS>, and <DEL> keys work as usual within the highlighted cursor areas.

After the data have been entered on the screen and any necessary corrections made to discovered errors, the user presses the <ESC> key. Now the following horizontal menu will appear at the top of the screen:

---Add---Reuse---Edit---Quit---.

The user must select the "Add" option to place the screen data into the database. The screen entry form will reappear, but it will be empty. By selecting the "Reuse" option the data is both stored and returned within the screen entry form. This enables the user to reuse any portion of the data for another separate entry if so desired. The

"Quit" option exits the user from the entry form without saving any data placed upon the screen. This is different than in editing and deleting procedures. DO NOT confuse the two. Remember to select "Add" before executing the "Quit" option. [Ref. 4: pp. 118-122]

E. SYSTEM BACK-UP AND DATABASE REPACKING

To protect the data entered into the FLYCLUB database, it is recommended that the user perform a System Back-up and Repack at least weekly. During this operation the database is copied to one of the previously formatted floppy disk (as discussed in section 3.B.3 "Other initialization preparations") and the hard disk is repacked. The hard disk is repacked in order to reclaim any unusable disk space which accumulated during the weeks operations. With each entry and removal of table data, some disk space is unavailable for use and could cause the user to eventually run out of usable disk storage space.

The initial database copy is made in case there is a system failure during the repacking operation, at which time the database could be lost. Upon completion of the repack the database is recopied, requiring less disk space due to the successful repack, so the user will have the most up-to-date version with all associated computer pointers. The back-up disks should be labeled and dated to ensure the latest version is used in case of system recovery operations.

To perform the System Back-up and Repack, the user need only to select option (4) from the End-of-the-Month Transactions menu and then follow the directions given on the computer screen. Remember, the operation requires a minimum of two preformatted disk before execution.

F. EXITING THE MNFC PROTOTYPE

To exit the MNFC MIS Prototype and return to the computer's operating system, the user selects option (7) from the Monterey Navy Flying Club Main Menu. Once selected, the MIS Prototype will "close" the FLYCLUB database, performing any remaining database loading; thereby, preventing any lost data. The user should not exit the system by turning the computer off. Data could be lost or incorrectly entered upon losing system power. If no further use is required of the computer, the power can now be safely turned off with no danger to the MIS database.

G. MAINTENANCE

Software maintenance encompasses many things, from correcting one small "bug" to an entire redesign and recoding of a specific module within the program. Before any source code is altered, the user should invest some time and effort in gaining an understanding of the processes required to plan, execute, and document any and all changes to a program code.

The various application modules of the MNFC MIS Prototype system are so integrated, any change must be thoroughly considered and properly documented or major errors could occur while running the system. The errors could, perhaps, not even be seen by the user. The system may appear to operate correctly, however, the data being stored, manipulated, and printed could be inaccurate or even lost.

As a minimum, the individual who is assigned maintenance responsibility for the MNFC MIS Prototype code should have a thorough knowledge of:

- Relational database structures
- Structured programming techniques
- Software configuration change documentations
- R:BASE 5000 coding language and constructs
- The current MNFC MIS Prototype source code
- The MIS Prototype's logical functions
- The operations and functions performed by the MNFC.

Before actual alterations are made to the prototype's source code, the maintenance supervisor should have a firm understanding of the system's logical functions. The logical functions must be understood prior to any manipulations of the MIS Prototype's physical functions.

The writer acknowledges that errors will be found and/or requirements orginially incorporated in the prototype will change. These events will necessitate the need for software maintenance. The code listed in Appendix E through Appendix H is commented to help the maintenance programmer and management to have a better understanding of each logical sequence of the program modules and their interfacing characteristics.

The average user of the prototype should NEVER attempt to alter the code in any fashion for any reason. Professional outside assistance is highly recommended for any new requirements to be added or to correct any discovered errors.

IV. AUTOMATED CODING VS MANUAL CODING

A. INTRODUCTION

Appendix E through Appendix H contains the source code for the latest version of the Monterey Navy Flying Club (MNFC) Management Information System (MIS) Prototype. The source code presented is a collation of program code written by the author and those lines produced by the automated code generator feature of R:BASE 5000 software, Application Express, and CDR. D. R. George, USN.

B. APPLICATION EXPRESS

1. Overview

As an automated code generator, Application Express was designed to assist the non-programmer in writing relational database applications in the R:BASE 5000 language. Application Express reduces the time required to implement a simple application since mastery of syntax, command structuring, and file formating procedures of R:BASE 5000 is not necessary prior to system development. By using a series of menu choices, Application Express can guide almost any user through the creation of a simple database structure, menu-driven entry and display forms, single table reports, and standard database processes. The user can start with a simple, straightforward application and add sophistication as the user's knowledge of both relational database structures and the R:BASE 5000 programming language increases.

2. Operations Available

From within the Application Express code generator the user is able to:

a. Define an application--establish the database structure
- name the database to be used
- name relational tables
- name table attributes
- define the attributes' data type

b. Change an existing database
- add tables
- delete tables
- rename tables and attributes
- redefine an attribute's data type

c. Cefine a new application
   - build a simple database program
   - define menu names
   - choose simple database options and actions
   - attach separately coded modules to the generated application

d. Change an existing application
   - add menu options and actions
   - delete menu options and actions
   - modify any menu, options, or actions
   - modify command module code

The basic database processes that are capable of being coded and implemented by the Application Express code generator include:

- load table data
- edit specific rows of data
- delete rows of data
- browse/review entire tables
- select/display rows of data
- print selective table data

Once the user has defined the database structures, developed and typed the menu listings, and completed the choice of options and actions, Application Express writes the program code, compiles the code, and combines the associated command files within one application program. The newly compiled program code is saved by Application Express in ASCII format and is listed as: <Application Name>.APP. The executable binary computer code file, labeled with the suffix of .APX, is also prepared by Application Express and is ready for immediate use.

3. Limitations Imposed

The full power of the R:BASE 5000 programming language is not available to the user while using the Application Express code generator. Database processing code produced by Express can be performed on only one table at a time. Multiple table inquiries, an advantage of a relational data management program like R:BASE 5000, can not be accomplished with Express generated code. Syntax and constructs for multiple table computations and summary reports (printing data compiled from more than one table) are not contained within Express.

Data entry and revisions are also limited to one table per operation. R:BASE 5000 is capable of variable entry (entering data into many tables from one screen entry form), and multiple table searches and revisions. However, Express entry forms are limited to handling only 20 columns or attributes, thus setting an upper boundary on the user's database structures. Searches coded by Express can sort one table with a maximum of three attributes and a single conditional clause. Reports printed from Express generated code can only contain data from a single table and are limited to a maximum width of 80 characters.

Another major limitation with Application Express is its inability to run individual modules. The user must conduct all debug procedures by executing the entire application created by Express from outside Express and in the R:BASE basic mode. As the program size increases, the time required to load the application into Express, make any changes, and then recompile the code, prior to testing, also increases considerably. All these limitations significantly reduce the capability of most relational applications and increase the difficulty of system implementation.

As with any code generator, the resulting source code from Application Express is no better than the database design plan that conceived it. The user must be very familiar with relational database theory, database structures, and programming logic to fully utilize Express efficiently. The application developed, using Express, must be thoroughly designed, accounting for data redundancy necessary to accommodate user searches, report printing, and database change constraints of the Express code generator. [Ref. 5]

4. Future Advancements

The writers of R:BASE 5000 series software, Microrim, have improved the R:BASE code generator process with the release of its R:BASE System V. System V consists of three programming features, Application Express, Forms Express, and Reports Express. With System V the user is offered:

- multiple form entry and editing
- multiple table entries
- Express assisted form and report generation
- enhanced report printing
- data validation
- multi-user (Network) capabilities
- greater inter-table relational operations
- improved mathematical computations [Ref. 6].

Even with all its new capabilities, R:BASE System V does not guarantee a bug-free application. The user still must properly design all database structures and logical processing steps. System V does eliminate the time to master the programmming syntax and constructs, and reduces the time from conception to implementation.

C. THE APPLICATION EXPRESS AND MANUAL CODING COMBINATION

Due to the anticipated size and complexitiy of the Monterey Navy Flying Club MIS Prototype, and the limitations associated with the R:BASE 5000 Application Express code generator, the writer chose to use Express to assist in producing the menu screens, in providing menu selection code, and in attaching the individual command files into one executable application program file. As can be seen in the Prototype's source code listed in Appendix E through Appendix H, all menu related code is computer generated by Application Express. Overall system execution time was reduced since the numerous command files were attached and compiled into one operating application file by Express. This allowed the reduction in the number of disk reads necessary to call and display each menu screen command file and eliminating the calling of another separate command file to interrupt and execute the user's responses.

Manual coding at the R:BASE command level was used to take advantage of the strong R:BASE 5000 relational database management language constructs. Multiple table searches, inter-table computations and comparisions, multiple table entries, multiple table editing, and integrated reporting were desired. The use of the relational database structure minimized data redundancy and reduced the number of required data structure tables within the database. To incorporate these capabilities, modular design and coding was required, which called for individual command files for each functional module and required the ability to test each module prior to system integration. Application Express did not provide the environment to handle these programming techniques, leading to its use only in menu code generation and application compiling. The system design with each major function (such as financial transactions), being coded manually and compiled under Express as separate applications, enhanced the prototype overall development and operation.

The critical function of module testing was conducted by coding each desired system function in a separate R:BASE 5000 command file. These files were compiled and run from within the R:BASE basic environment. This allowed for error discovery, correction, and retesting prior to placing that module into the major prototype application. Using the Express option "Macro", each command file was incorporated into the Express generated system.

However, the ease with which Application Express structures and builds database tables proved to be an excellent asset to the writer. Additions, revisions, and deletions of table structures and attributes were conducted exclusively from Express. The user-friendly menus and prompts leads the user through the process of table definition and data type definitions.

The basic R:BASE Forms and Reports coding procedures were used in order to meet the system requirements for multiple table computations and reporting. All screen entry forms and report printing commands are located in the R:BASE database structure as a table. When entry and print commands are encountered within the source code, the application calls the format desired, whether for entry or printing, from the already accessed database. This concept also assisted in reducing the time required to produce the product since separate disk reads are not necessary to locate the commands to be executed.

V. CONCLUSIONS AND RECOMMENDATIONS

A. SUMMARY

Version 2.0 of the Monterey Navy Flying Club (MNFC) Management Information System (MIS) Prototype was successfully designed, developed, and implemented in this thesis. In continuation of an earlier thesis written by CDR. D. R. George, the original prototype was expanded to include the remaining administrative and financial requirements of the club which were not previously incorporated [Ref. 1: p. 9]. This completed MIS prototype can be used as a springboard from which to investigate the possibilities of Decision Support System (DSS) modules to assist in the decision-making processes of the Club Manager and the Board of Directors. The prototype was implemented in R:BASE 5000 database language, a product of Microrim of Belleview, Washington.

The Monterey Navy Flying Club's Management Information System (MIS) Prototype is an interactive, menu-driven system consisting of five separate, functionally independent, applications (programs) which access one relationally structured database. Each application gives the manager an easily accessible tool to manage club membership, process member charges and billing, track aircraft maintenance and associated costs, and disseminate information to governing authorities. The system provides the framework from which a Decision Support Package application can be designed and implemented to afford the Club's Manager and Board of Directors greater decision-making support.

The current MIS Prototype source code is a combination of manually written R:BASE 5000 command files and computer generated code. Limitations imposed upon the use of special command syntax and constructs and module testing by R:BASE 5000 Application Express (the code generator) lead to the exclusive use of Express for all menu screen and selection coding, application interface coding, and application structuring (the integration of individual command files into one program) and compiling. Functionally distinct modules were coded manually in the R:BASE 5000 command language to take full advantage of R:BASE 5000's relational database capabilities. Multiple table integration, tabulation, searches, entries, and independent module testing were available when the code was written outside of the Express environment.

The development of the relational database tables was easily and rapidly accomplished from within Application Express. Minimum keystrokes were necessary to change existing table structures or attributes while in Application Express. However, source code comments are not provided for lines of code written by the code generator. This deficiency makes it very difficult to maintain the code.

The User's Guide, Chapter III, and version 2.0 of the MIS prototype is undergoing system review and operational testing by a potential user. Outside review and testing of the User's Guide will ensure its ease of reading and clarity. System run-time errors will be documented and corrected.

B. PROJECT RESULTS

The Monterey Navy Flying Club's Management Information System (MIS) Prototype produces various hard copy products. Each report or statement is in response to either an administrative or financial requirement. These products provide a synopsis of club activities which are utilized to estimate future budgets or explain current financial conditions. The reports and statements currently prepared by the prototype's latest version for management's review or club distribution are:

- Membership Roster
- Monthly Manager's Report
- Aged Accounts Receivable Report
- Member Statements
- Instructor Statements
- Lessor Statements
- Club Balance Sheet
- Club Income Statement
- Operating statement inputs to the Annual Report
- Summary of Aircraft Insurance
- Aircraft Inventory and Status Report
- Membership Breakdown Summary

All database data are saved in historical files providing depository of data from which to make future inquiries and offering greater opportunities to support decision-making applications. The club's monthly transactions are saved within the database by date, account, and categorized by function (e.g. accounts payable payment is listed in the accounts payable paid table). Monthly, all accounts are posted and billing is conducted. Upon the completion of these end-of-the month transactions, the revised database is saved on a floppy disk. Each month's transactions are appended to the previous month's historical file. Thereby, each month's floppy disk carries a snapshot of the database from the beginning of the year to that month. System recovery can then be accomplished by copying the latest floppy disk and by entering those transactions which occurred since the previous month's posting.

C. SUGGESTIONS FOR FUTURE RESEARCH

This thesis successfully designed and implemented a MIS prototype system for the Monterey Navy Flying Club. Conceptionally, the original prototype system was to possess a limited Decision Support System (DSS) package capable of accessing the relational database and assisting management decisions. The accompanying DSS package was not attained. The current system design structure will allow the addition of this future DSS application program to access the relational database as conceived.

1. System Implementation and Future Maintenance

Over the life of the MNFC MIS Prototype, outside influences will cause software maintenance to occur. Newly discovered errors, new system procedures, revised reporting reguirements, entry/edit format changes, and priority shifts by higher management are inevitable. These events will force the need to make system modifications to the existing prototype. To accommodate these necessary actions, it is recommended that a maintenance supervisor be contracted to conduct any system analysis, design, coding, and debugging of the prototype.

This maintenance person would be responsible for data structure control, database documentation, configuration change control, enhancement evaluation, and recovery procedures. He should implement a simple method for the user to document all errors and enhancement requests. The prototype's database will require considerable care. The maintenance supervisor should be considerably experienced in database processing activities, familiar with the club's operations, and able to integrate existing systems with the prototype. Club management is responsible to keep the maintenance supervisor informed of all regulational changes and additions.

It is recommended that the MNFC MIS Prototype be installed by the club and tested in parallel with the current manual system for a minimum of two months. All system errors discovered during the running of the prototype should be documented for use in subsequent program debugging. Any user modifications should also be specified during this system test period. Application enhancements then can be provided to the system maintenance supervisor for analysis, design, and implementation. Approximately two months should be allotted for initial application corrections, enhancements, and retest, followed by system turnover to prototype operations.

Presently, there is no capability to track individual aircraft parts purchased and expended by the club's manintenance personnel. Due to the lack of this data, inaccurate estimates of future maintenance cost may be produced. Cost trends for specific parts and usage trends for those highly critical aircraft parts cannot be provided. It is recommended that the prototype be enhanced to include an aircraft inventory module to maintain the club's aircraft parts inventory. Overall maintenance cost per aircraft is already kept by the system, but individual part identity and cost could be useful data to formulate better management and control of overhead cost.

It is also recommended that the club manager and the maintenance supervisor determine the procedures to clear the prototype database and prepare for a new year's operation. Procedures must be established to clear specific database tables while maintaining tables, such as the BAL_SHET, EARNINGS, ACCT_REC. The size of the database will become too large for retention on a few floppy disks unless some data are stored elsewhere before continuing with club activities.

2. Possible Decision Suppport

It is recommended that a continuation of this thesis be undertaken to design, develop, and implement a Decision Support System (DSS) package for use with the FLYCLUB relational database. Initial analysis indicates that a combination of R:BASE Clout software, interfacing programs, such as Multi-Plan and Wordstar, and the basic R:BASE 5000 inquiry syntax and constructs demonstrate promising possibilities for decision support. The ease with which R:BASE data can be transferred between programs such as Multi-Plan and Wordstar offers the user the ability to manipulate data and present results to interested parties in various formats.

In particular, a Decision Support System package could be used to assist the Monterey Navy Flying Club Manger and its Board of Directors in their strategic planning. Future budgetary plans would provide more accurate data of past expenditures. Individual aircraft profitably could be investigated and consideration be given to the necessary cost to retain club aircraft. Trends in membership, maintenance, and operating cost could be tracked, analyzed, and be the basis for various forecasts.

The DSS application could be easily incorporated into the menu selection framework of the initiating application, FLYSTART. The DSS application (program) should provide access to the tools necessary to address the following major issues:

a. Predicting aircraft availability
   - Tracking scheduled maintenance for individual aircraft vs. expected club use
   - Predicting reoccurring types of maintenance for an aircraft
   - Tracking receipt of critical aircraft parts and their cost
   - Maintaining high usage parts on hand
   - Investigating reoccurring member associated maintenance

b. Budgetary forecasting
   - Predicting future rental revenues based on past flight hours flown
   - Comparing budgeted expenses with actual expenditures
   - Evaluating individual aircraft operating cost vs. revenue the plane earns
   - Estimating the usage of flight supplies by club and members
   - Maintaining flight supply stocks and cost within desired target

c. Membership trend analysis
   - Investigating membership rental trends for specific aircraft types
   - Reporting changes in membership flight certifications
   - Estimating future flight training requirements of current membership
   - Displaying instructor to student ratios
   - Forecast instructor availability for obtaining future certifications

d. Aircraft replacement planning
   - Comparing cost to buy vs. lease of aircraft
   - Anticipating avionics requirements to meet club needs
   - Targeting aircraft for replacement due to ops/maintenance history
   - Forecasting new specific aircraft suites
   - Evaluating cost to retain vs. replace a specific aircraft

e. Accident rates
   - Comparing pilot skills and type of accident
   - Displaying those members with reoccurring accidents
   - Searching for common errors/events associated to accidents
   - Identifying potential members for reoccurring accidents

This thesis was the second step forward for the Monterey Navy Flying Club in gaining firmer management control over club data. The prototype use and eventual modifications to the program structures will certainly reinforce the positive aspects of using computerized information systems. The addition of a decision support application will further that positive atmosphere with the club's Board of Directors and its Manager.

APPENDIX A
DATA FLOW DIAGRAMS

A Data Flow Diagram (DFD) partitions a system, and is one of the principal tools of the structured specification. Through the use of this simple graphic representation, the user and the analyst can examine the active components of the system and the associated data interfaces. DFD's are not intended to show control or procedural sequences as would a flowchart, but are to exhibit a system in small concise logical segments and to demonstrate each major data flow interface between the processes involved [Ref. 7].

Figure A.1 through Figure A.25 are logical DFD's and depict the essential functions of the Monterey Navy Flying Club Management Information System prototype. Each process, depicted by a circle, is uniquely identified by a number such as, 1.1 for Maintain Club Records in Figure A.2. The DFD process numbers are the same as the numbers assigned the major MIS processes in the System Hierarchy Charts shown in Figure C.1 through Figure C.3. Subprocesses are numbered such that one can readily see its root process. For example, the subprocess, Enter/Edit Member Record is assigned the number 1.1.1. The '1.1' digits note the root process, Maintain Club Records, while the '.1' tells the reader that this is the first subprocess under Maintaining Club Records. The subsequent subprocesses involved in Enter Edit Member Records will be noted by an extention of the '1.1.1' number in the same manner (e.g. 1.1.1.2 for Enter a New Member Record).

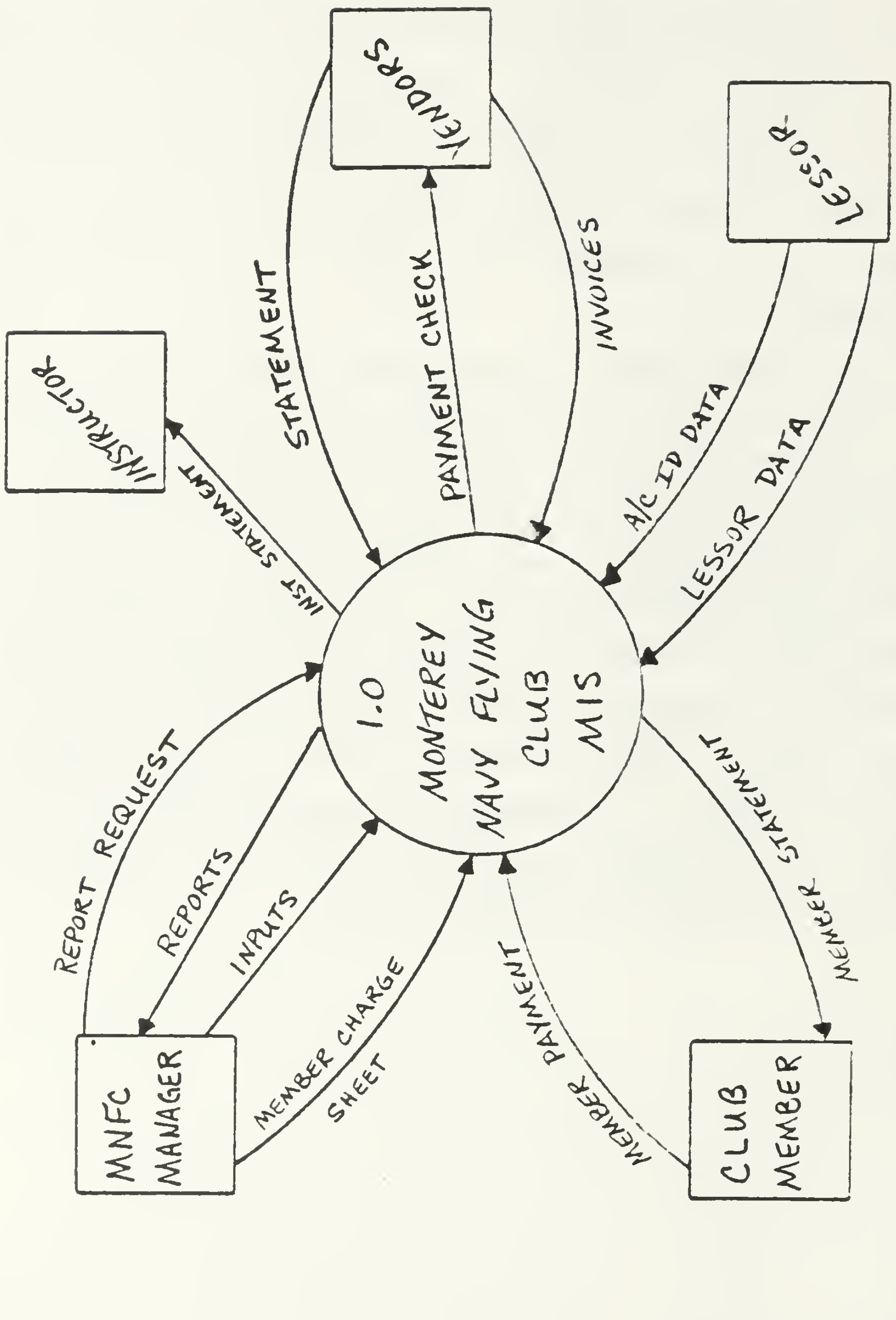


Figure A.1 Monterey Navy Flying Club MIS (Level zero).

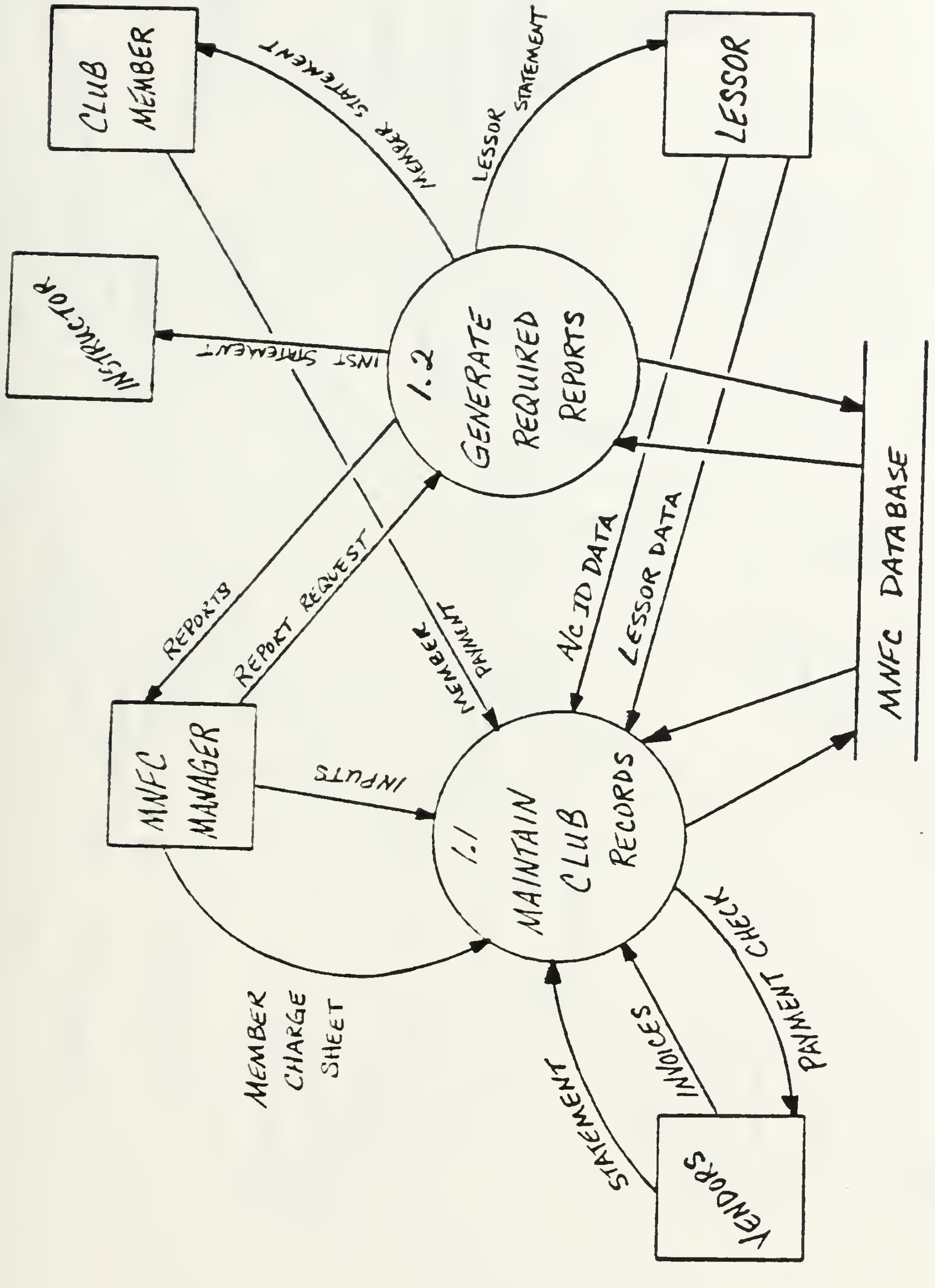


Figure A.2 Monterey Navy Flying Club (Level one).

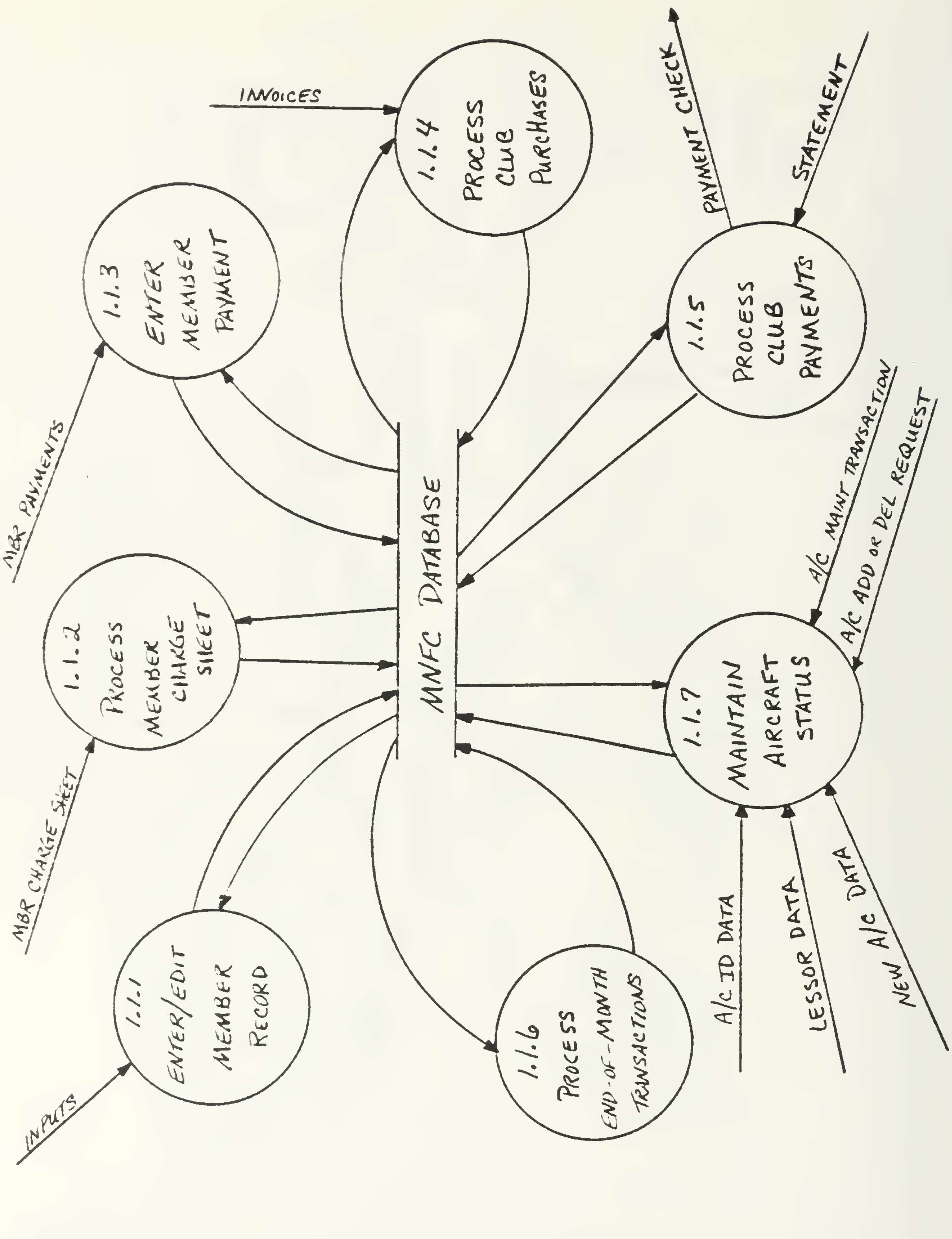

INPUTS
1.1.1 ENTER/EDIT MEMBER RECORD
MBR CHARGE SHEET
1.1.2 PROCESS MEMBER CHARGE SHEET
MBR PAYMENTS
1.1.3 ENTER MEMBER PAYMENT
INVOICES
1.1.4 PROCESS CLUB PURCHASES
MNFC DATABASE
1.1.5 PROCESS CLUB PAYMENTS
PAYMENT CHECK
STATEMENT
1.1.6 PROCESS END-OF-MONTH TRANSACTIONS
1.1.7 MAINTAIN AIRCRAFT STATUS
A/C MAINT TRANSACTION
A/C ADD OR DEL REQUEST
A/C ID DATA
LESSOR DATA
NEW A/C DATA

Figure A.3 Maintaining MNFC Records.

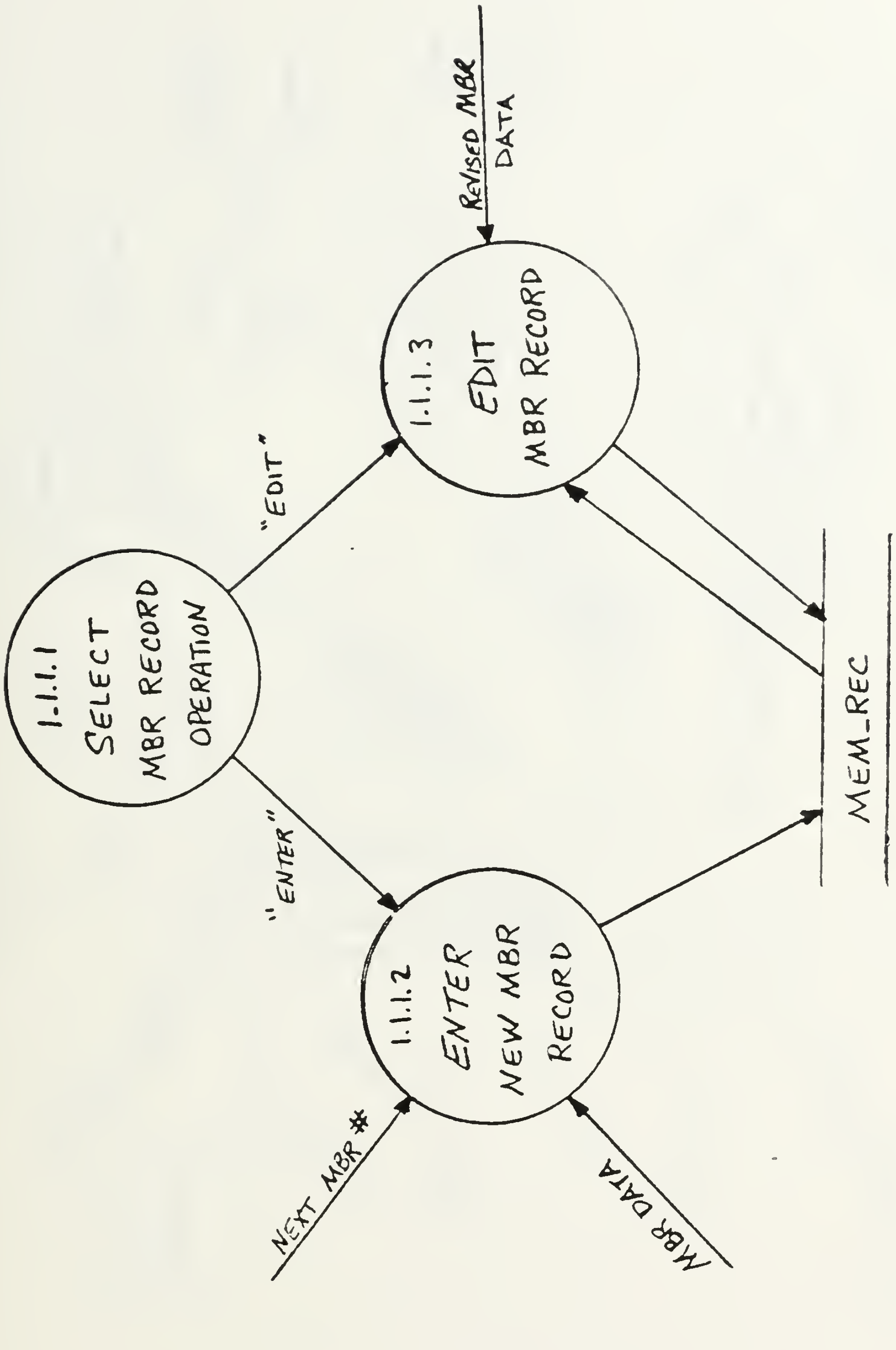


Figure A.4 Enter and/or Edit Member Records.

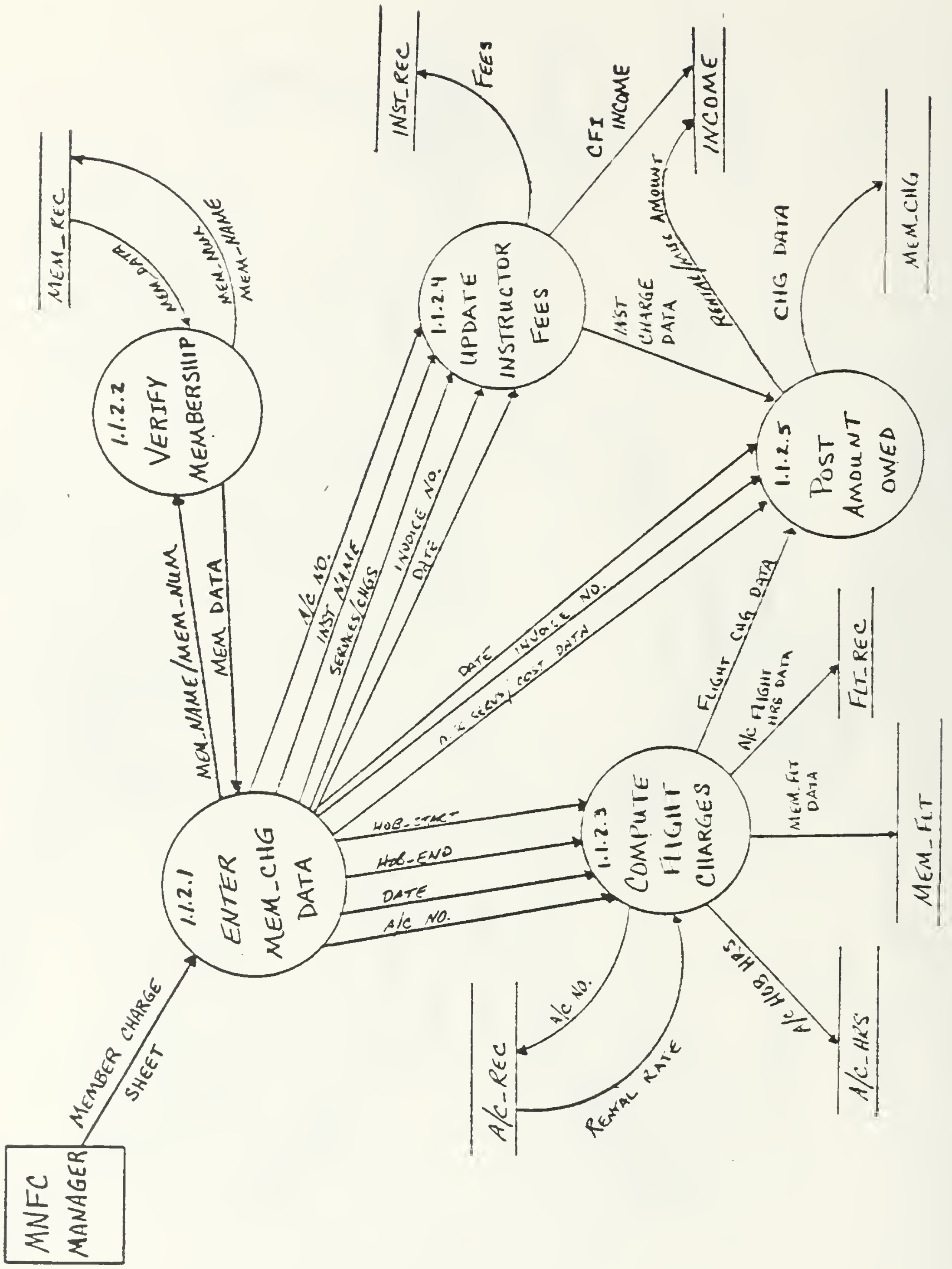


Figure A.5 Processing Member's Charge Sheets.

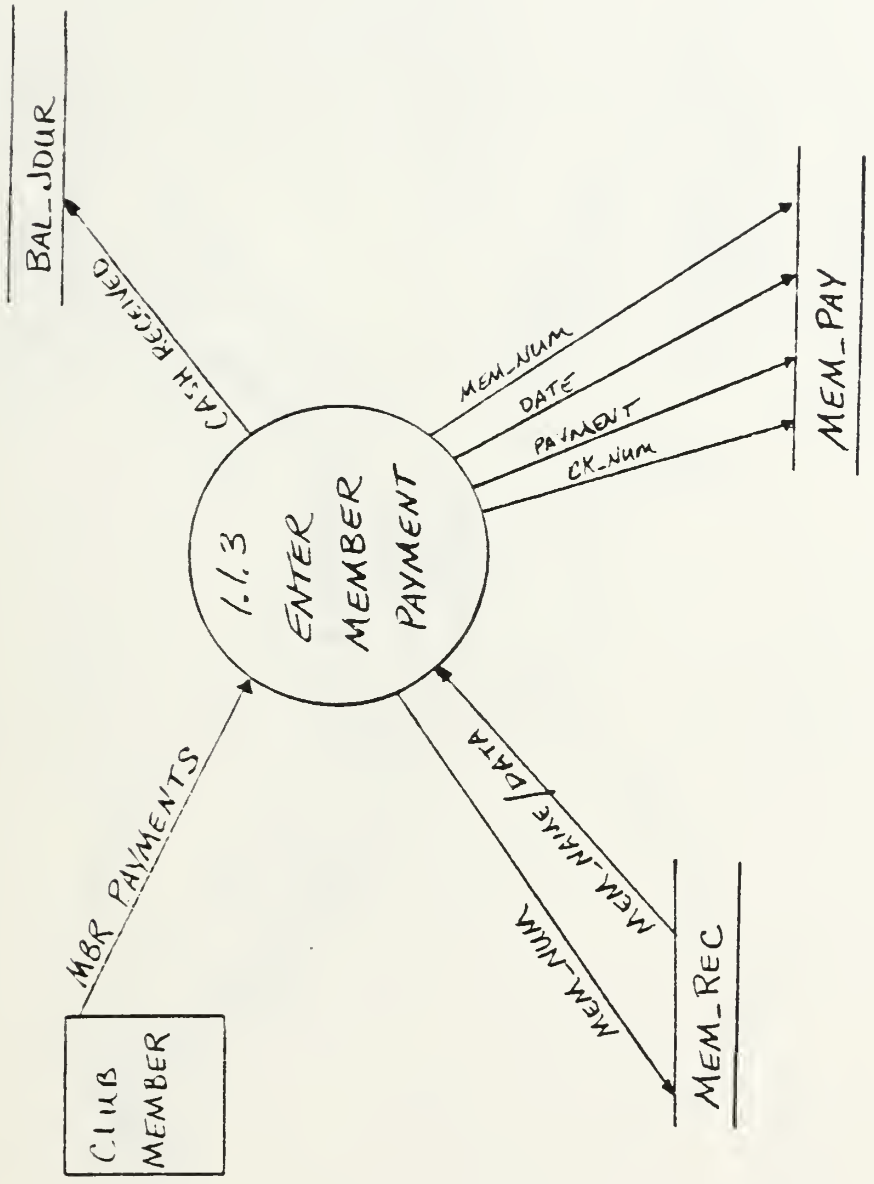


Figure A.6 Enter Member Payments.

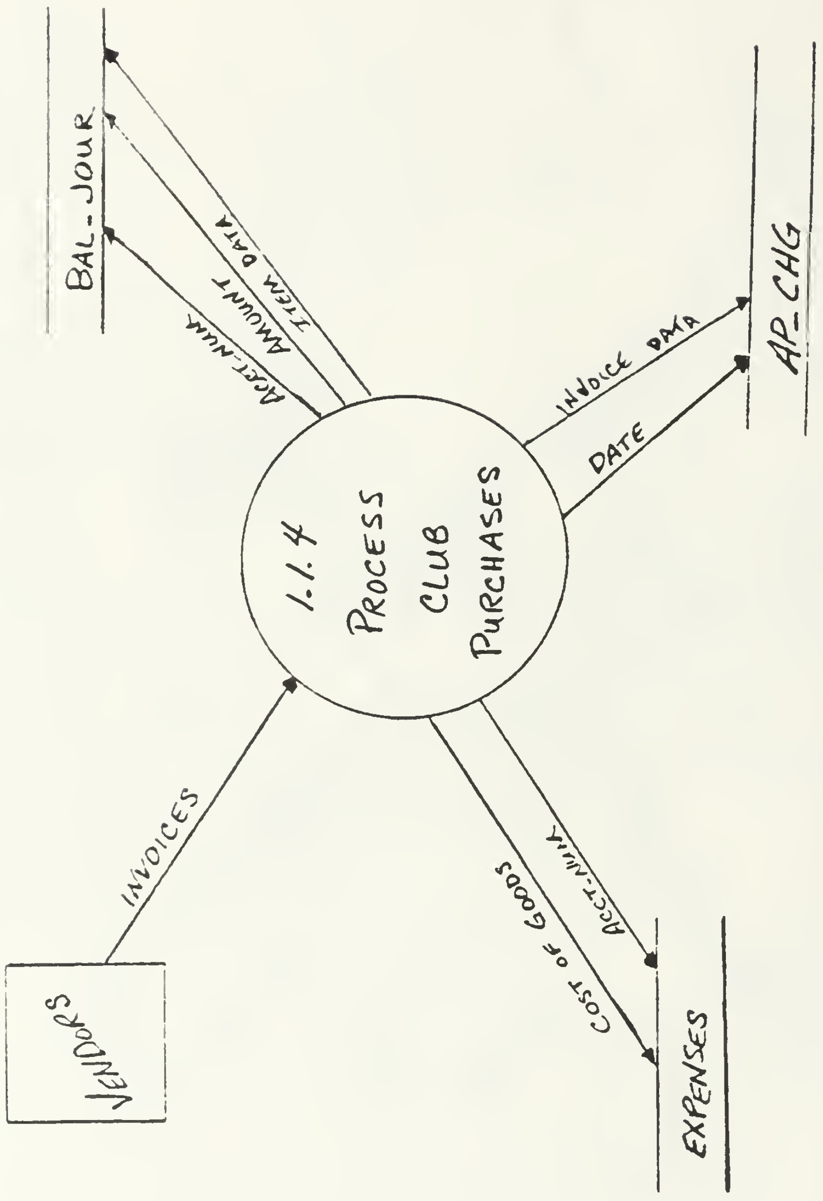


Figure A.7 Processing MNFC Purchases.

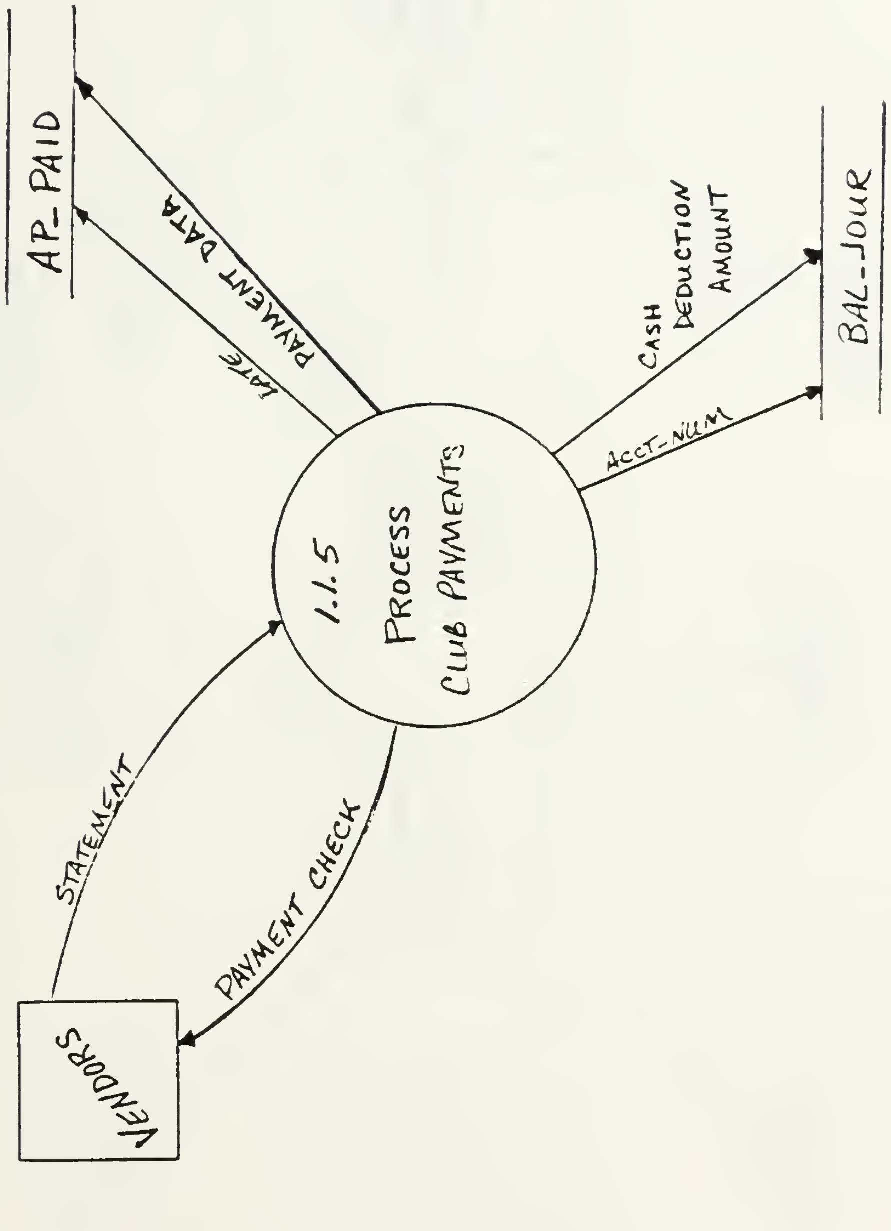


Figure A.8 Processing MNFC Payments.

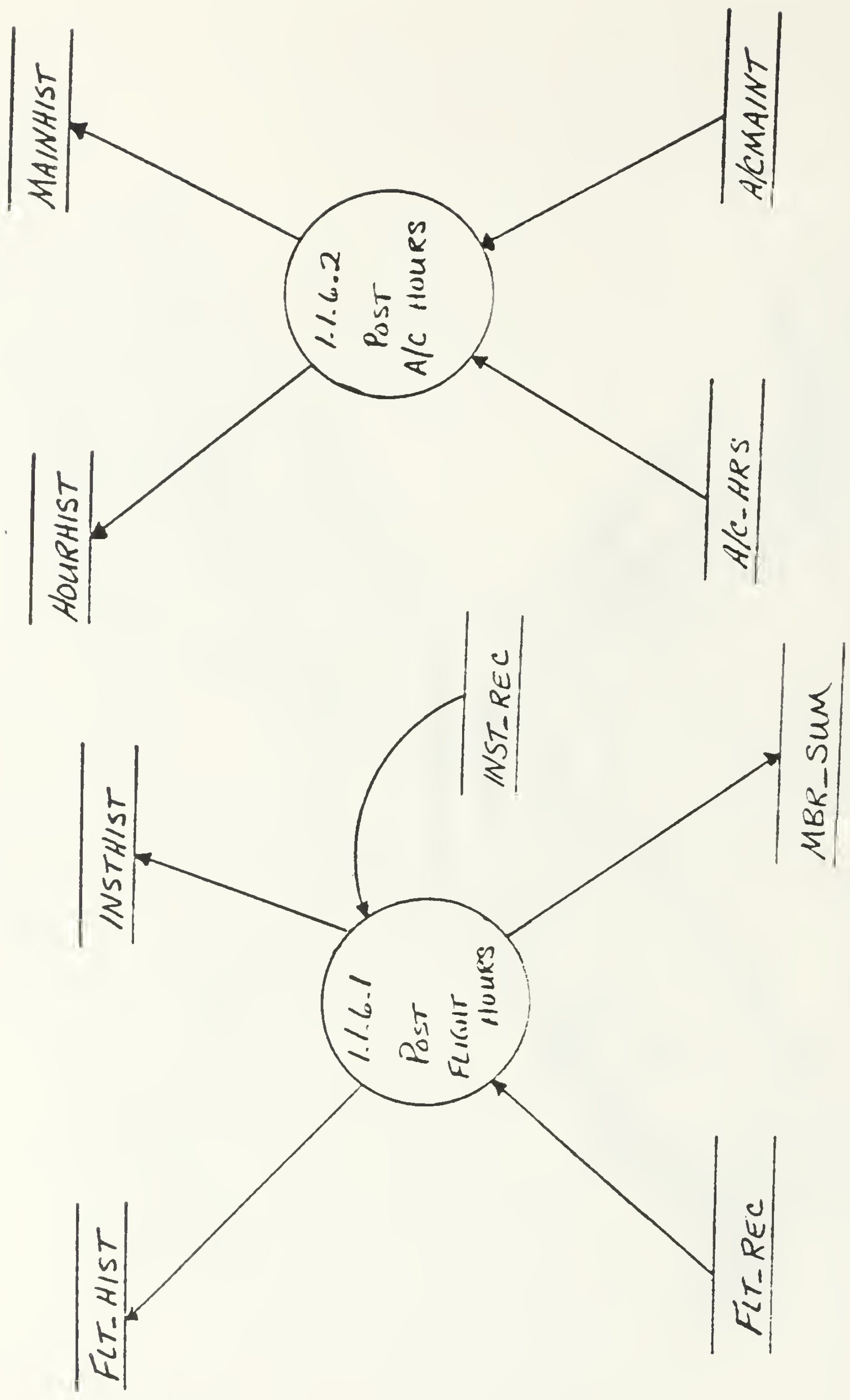


Figure A.9a Processing the End-of-the-Month Transactions.

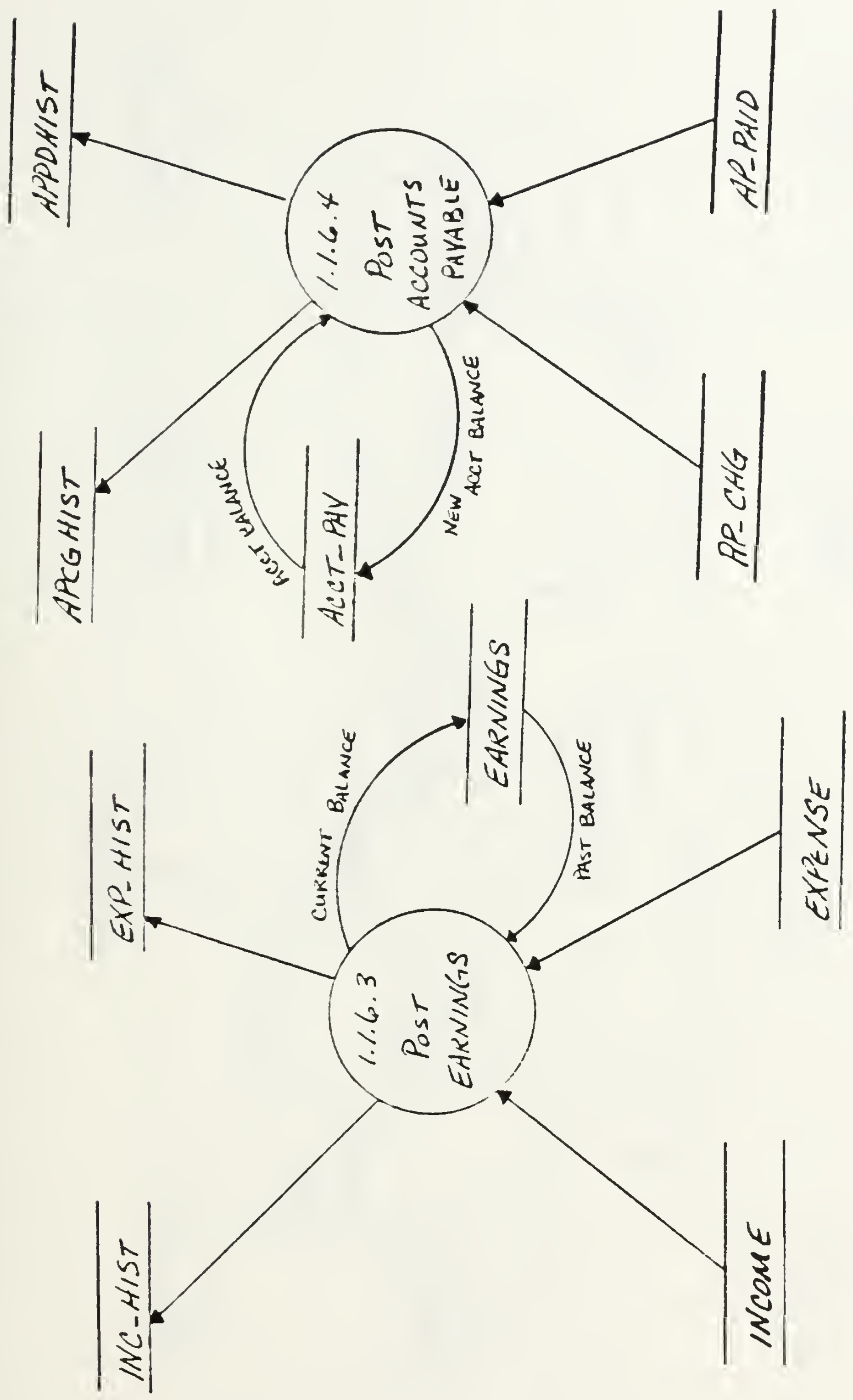


Figure A.9b Processing the End-of-the-Month Transactions (con't).

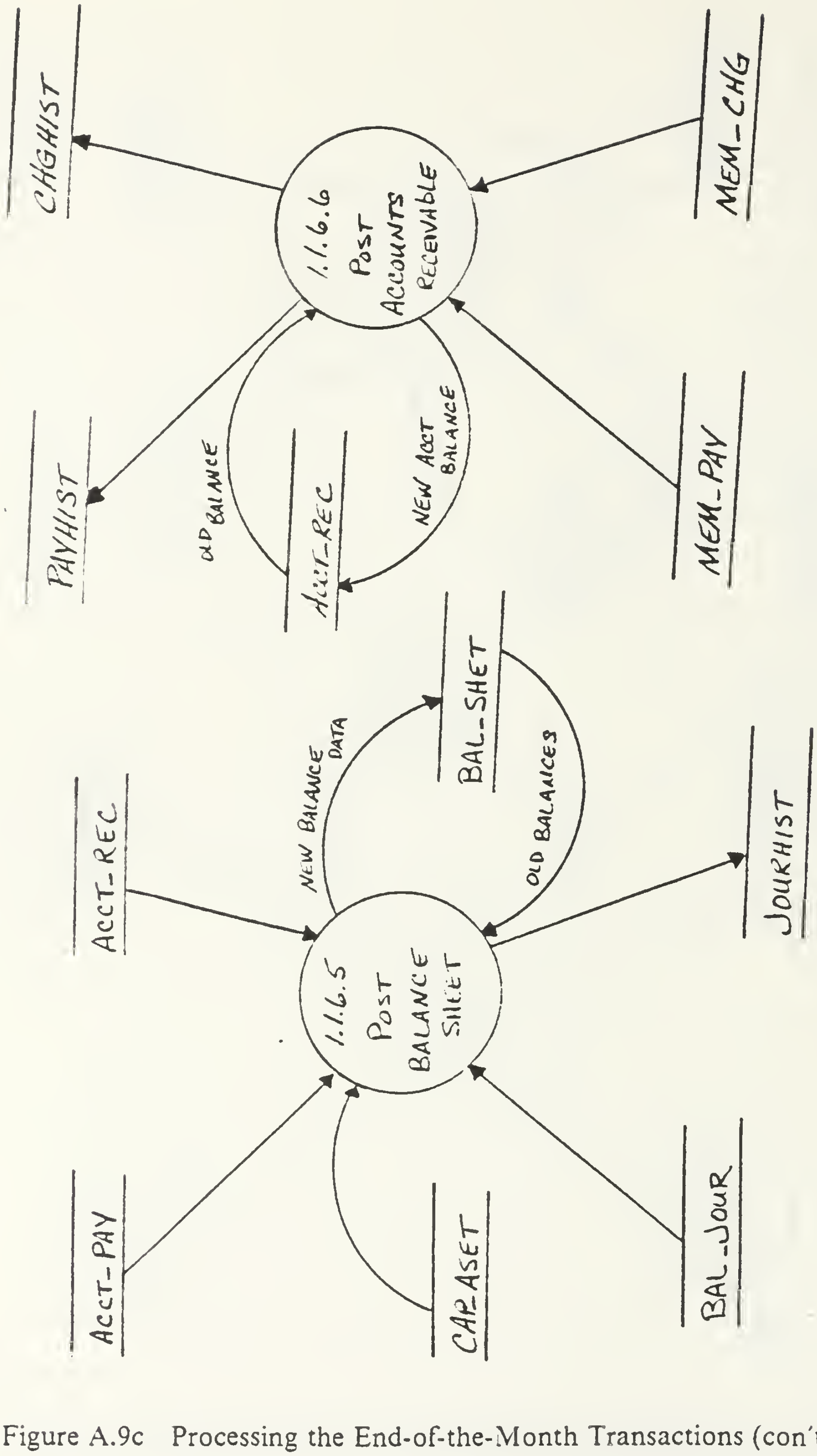


Figure A.9c Processing the End-of-the-Month Transactions (con't).

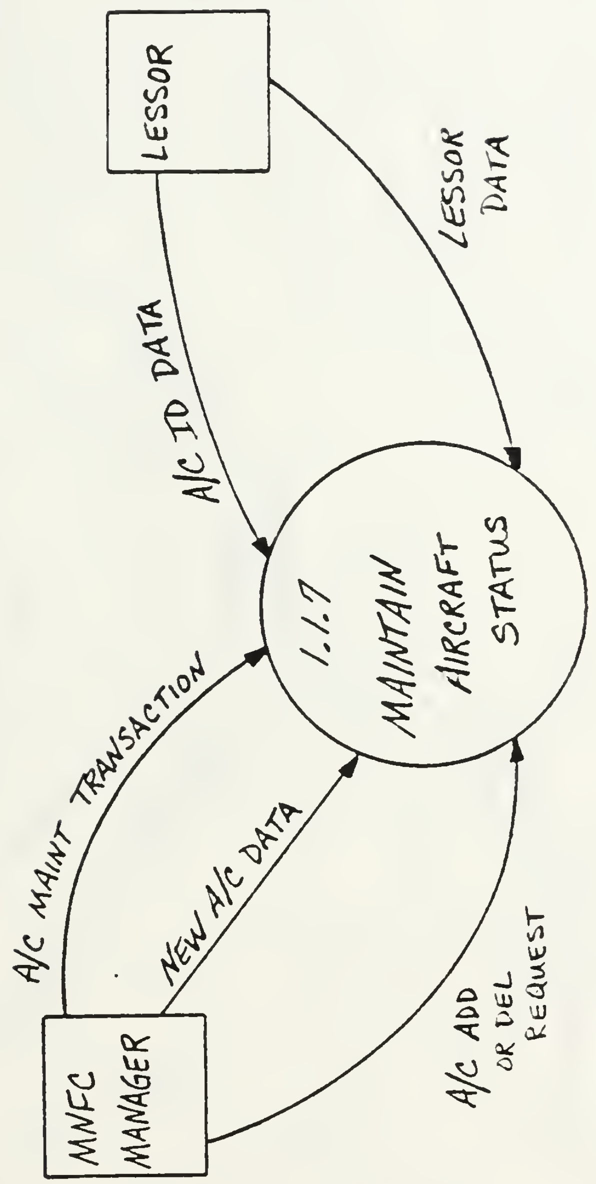


Figure A.10 Maintaining Aircraft Inventory and Status.

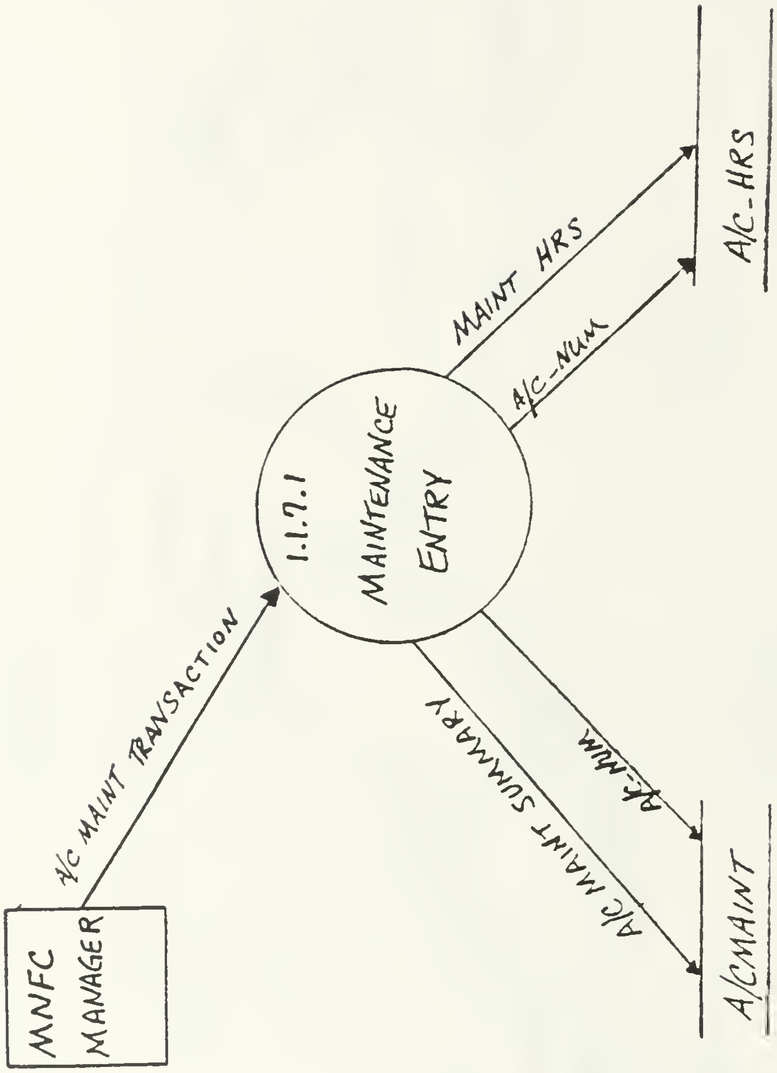


Figure A.11 Aircraft Maintenance Entry.

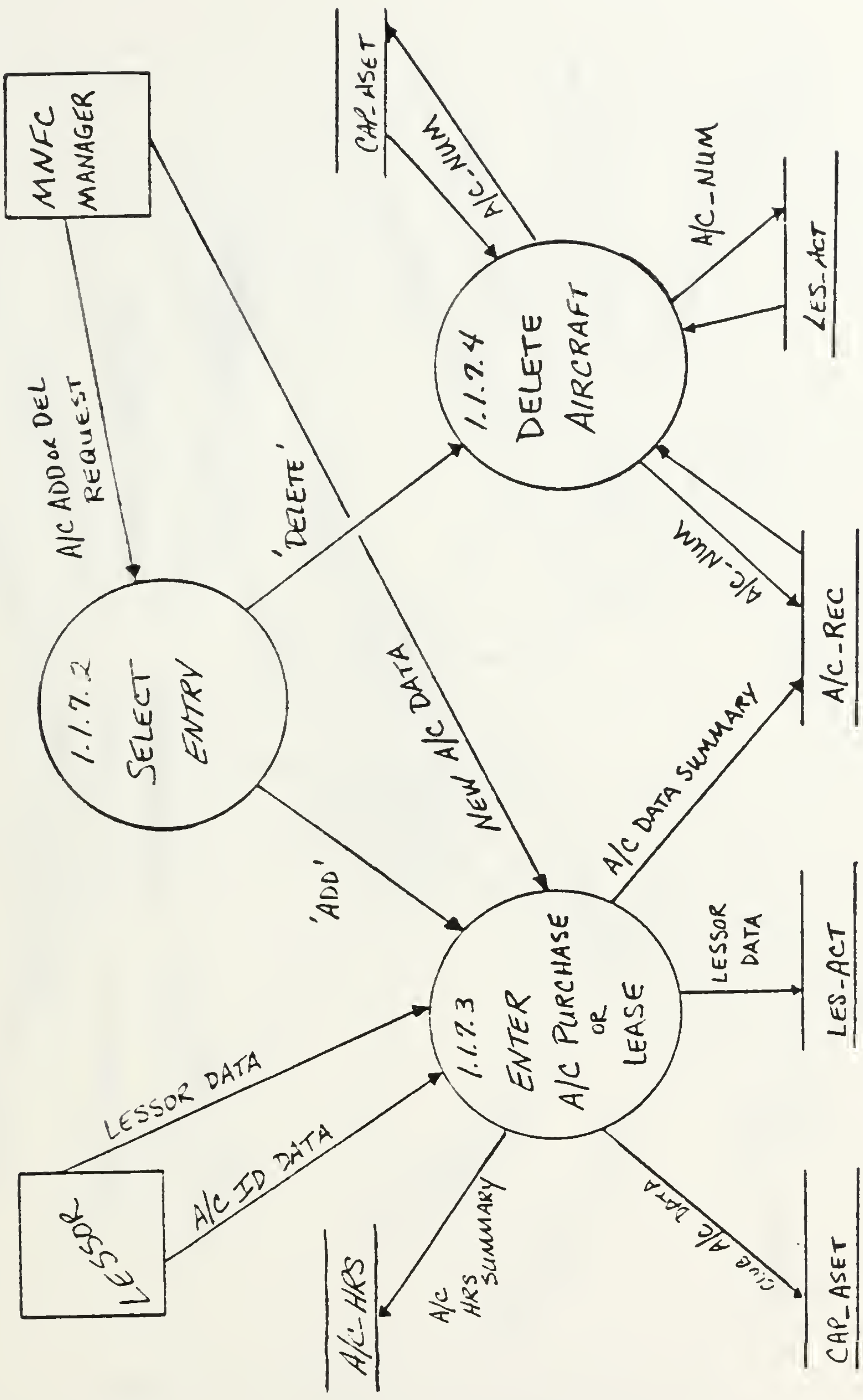


Figure A.12 Additions and Deletions to Aircraft Inventory.

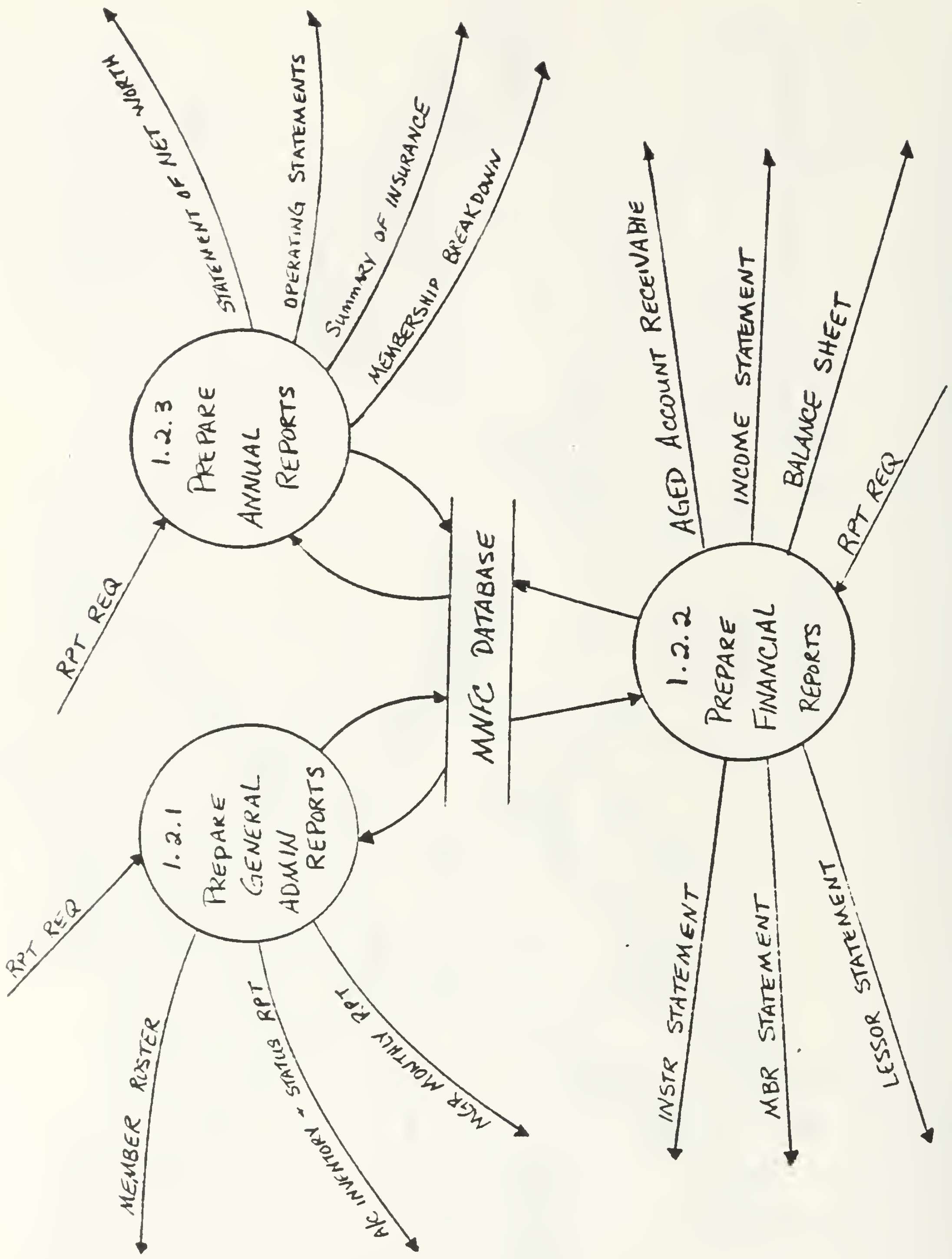


Figure A.13 Generation of Reports and Statements.

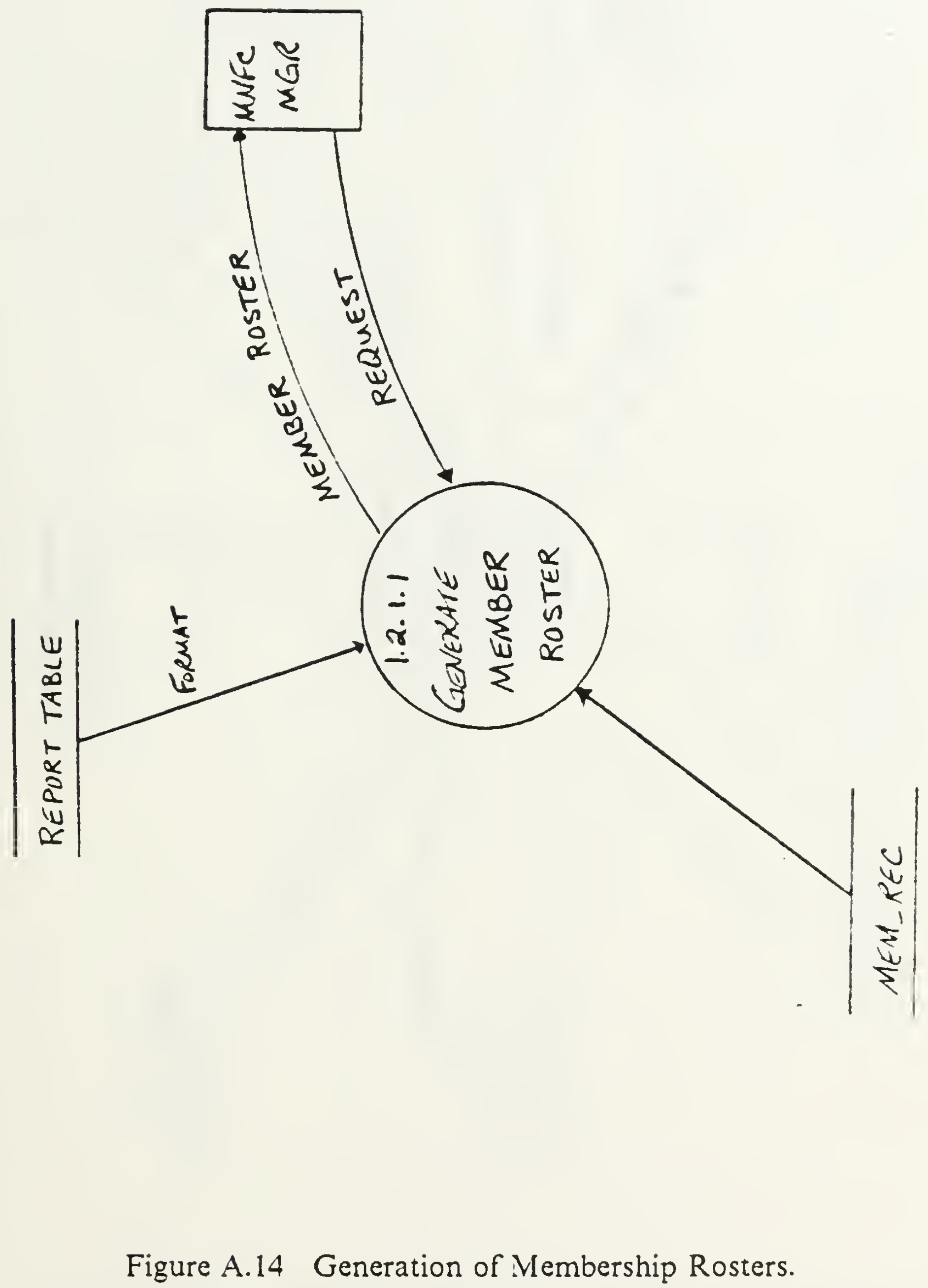


Figure A.14   Generation of Membership Rosters.

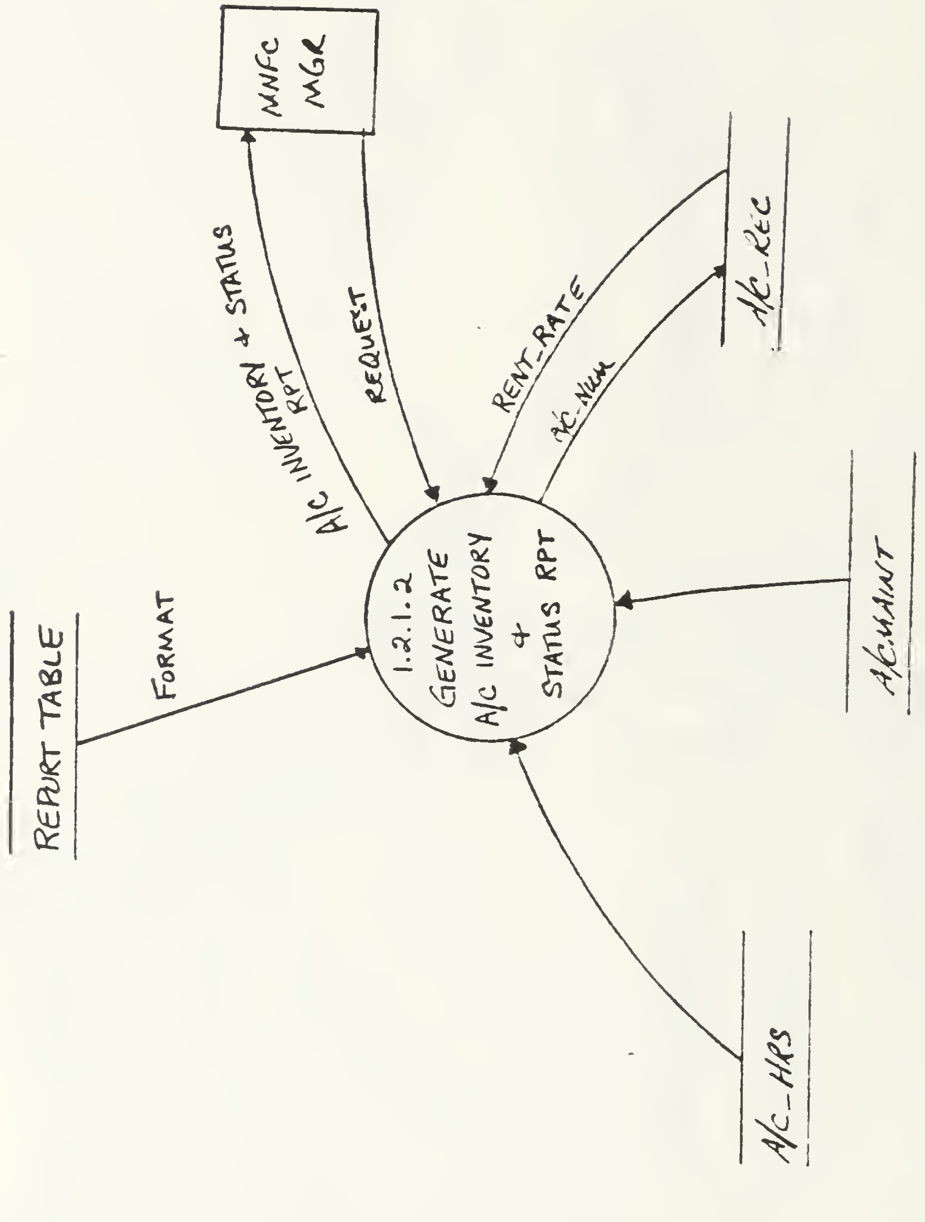


Figure A.15 Generation of Aircraft Inventory and Status Report.

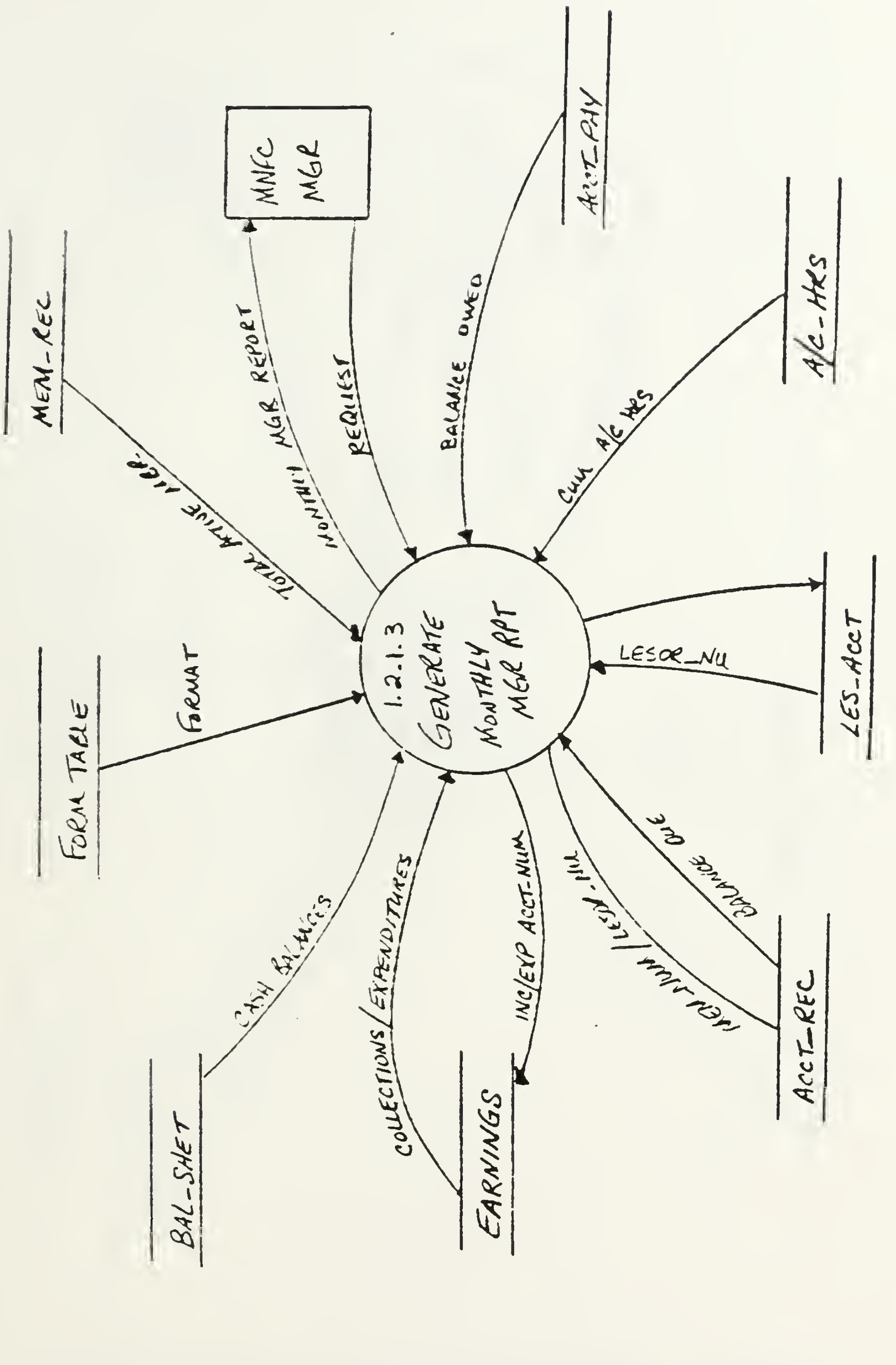


Figure A.16 Generation of Monthly Manager's Report.

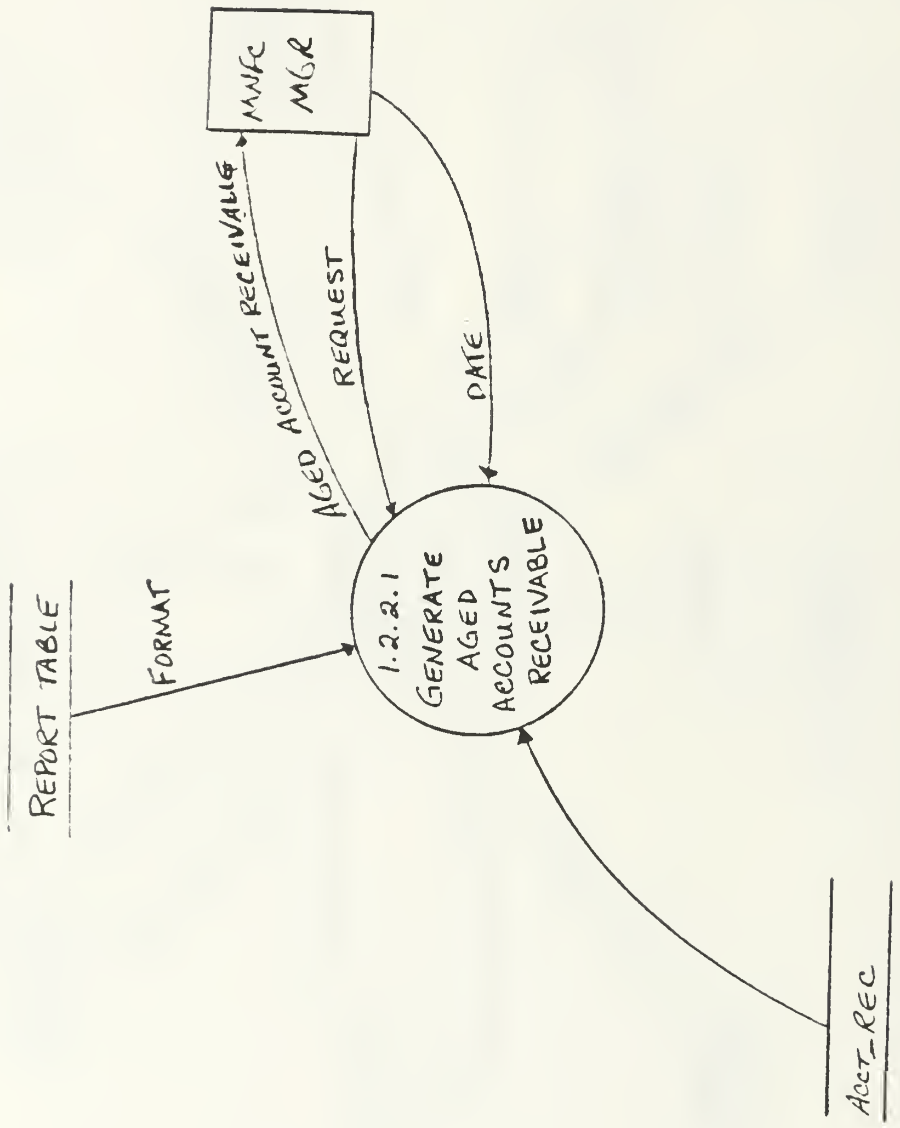


Figure A.17 Generation of Aged Accounts Receivable Report.

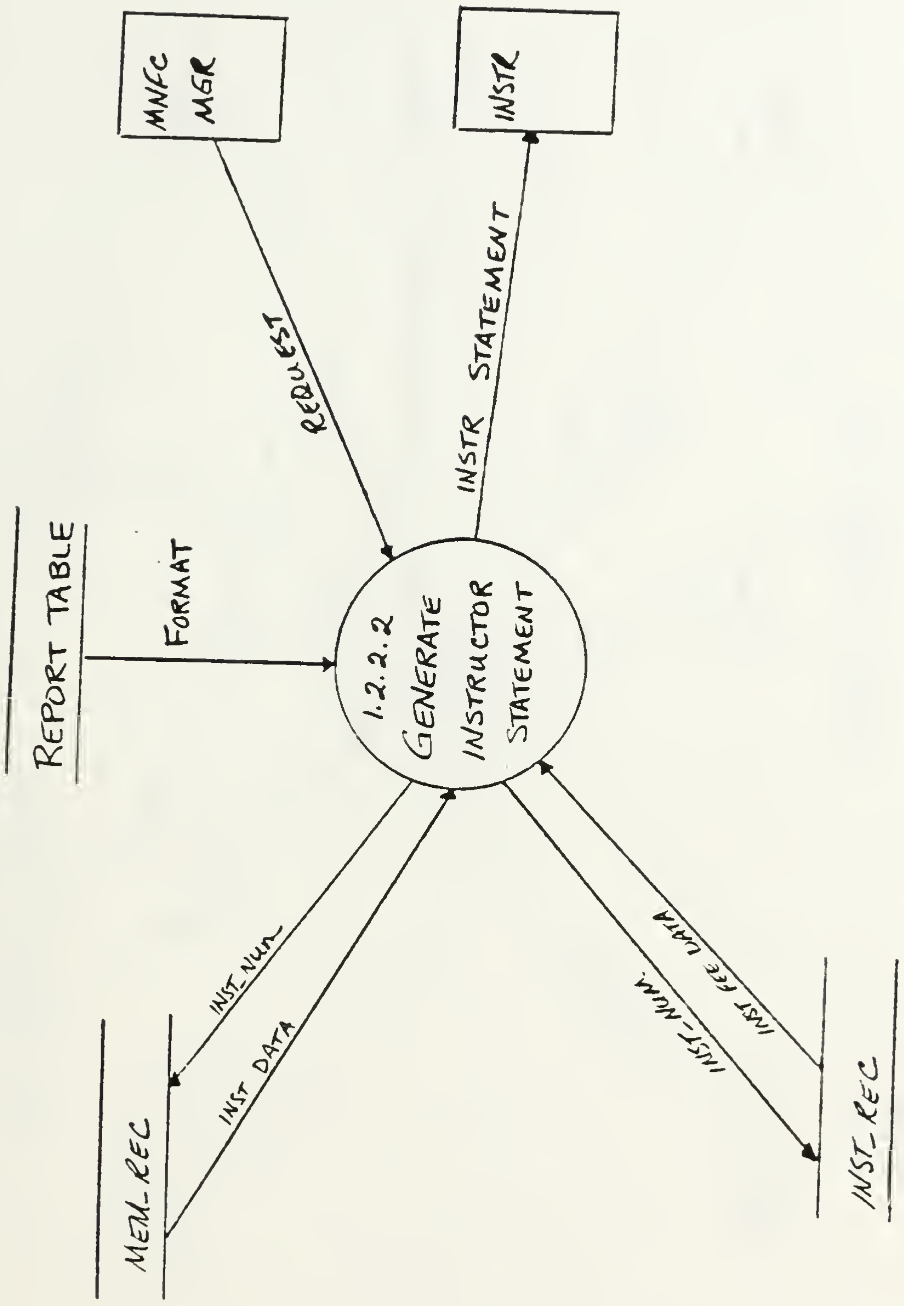


Figure A.18 Generation of Instructor Statements.

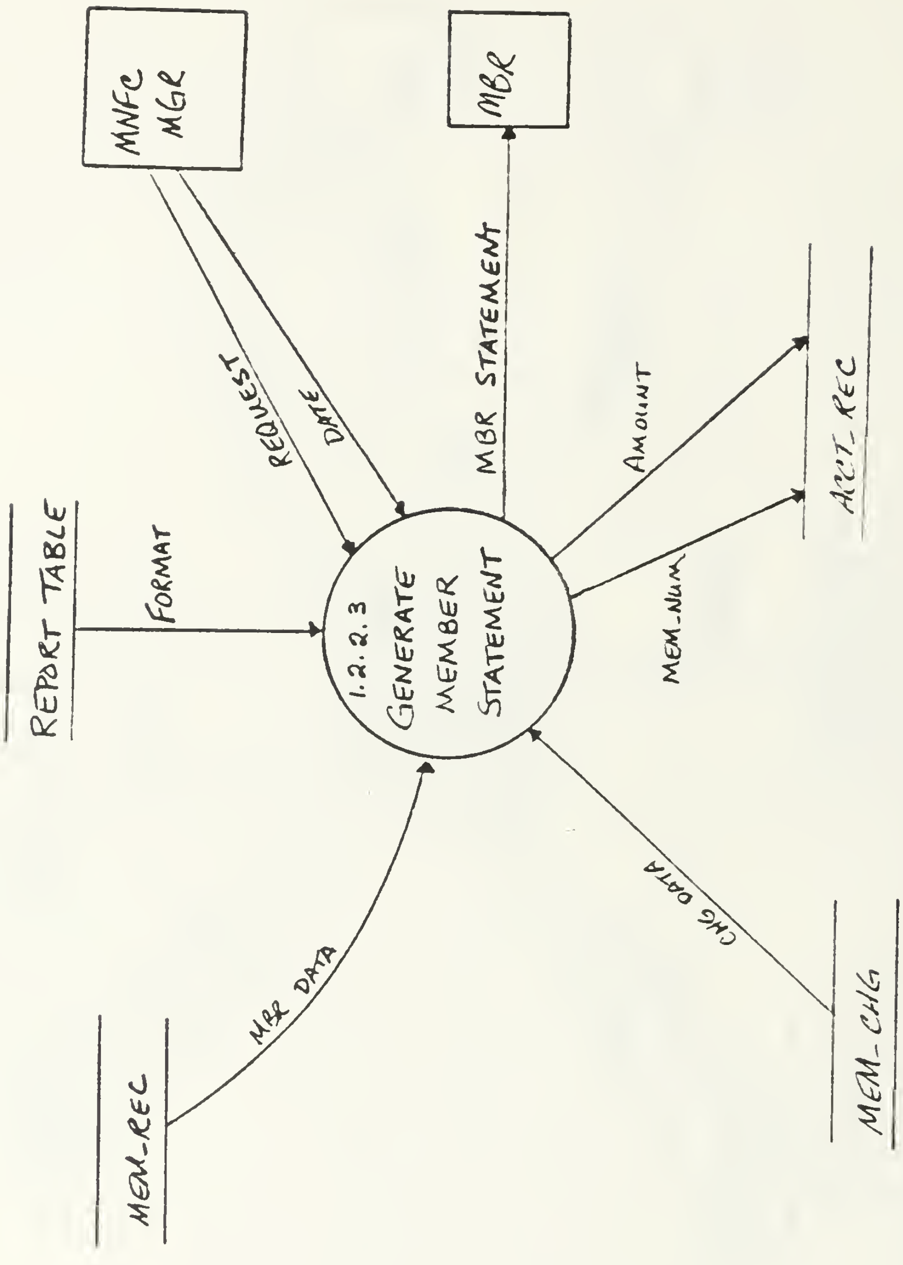


Figure A.19 Generation of Member Statements.

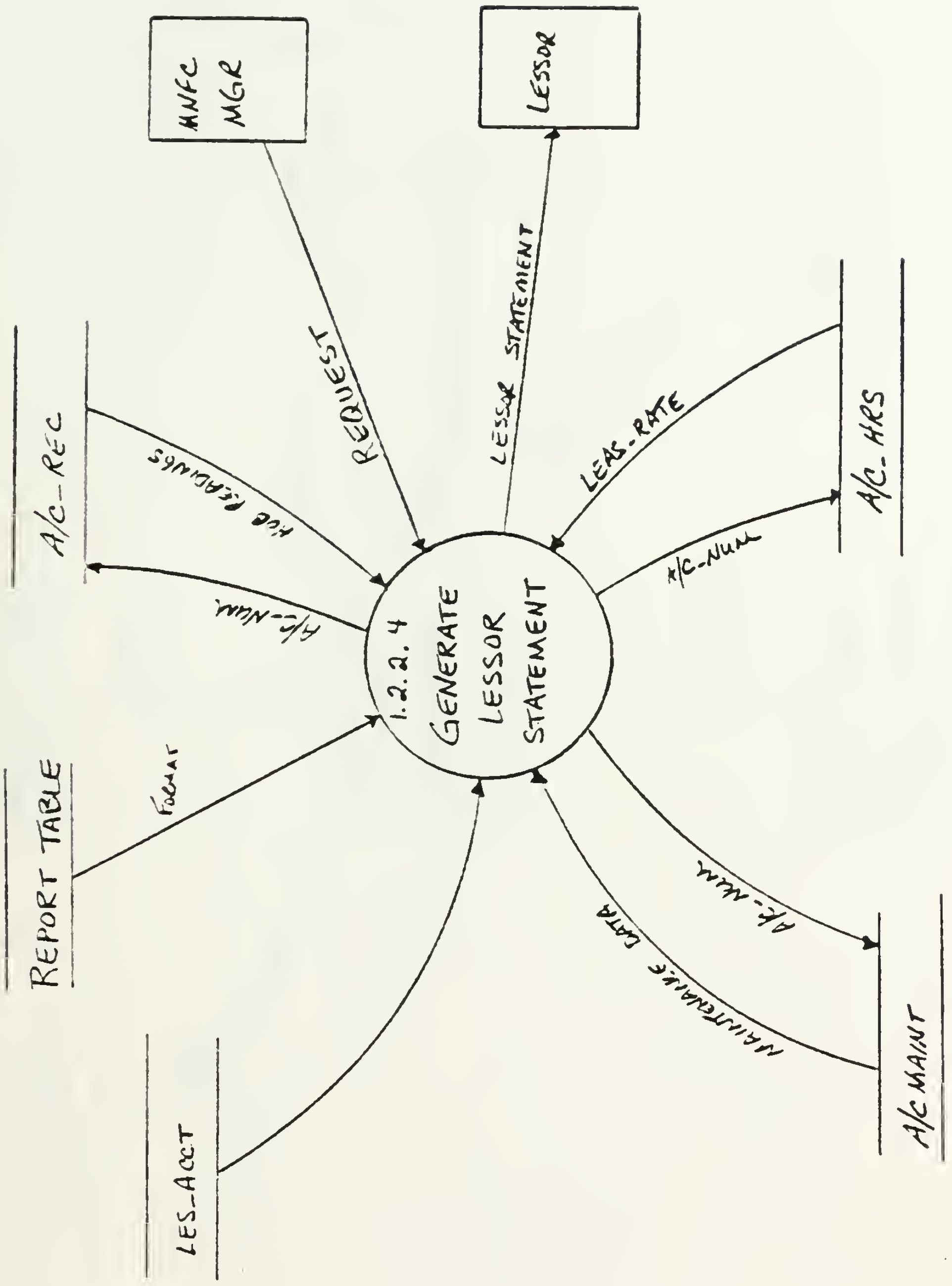


Figure A.20 Generation of Lessor Statements.

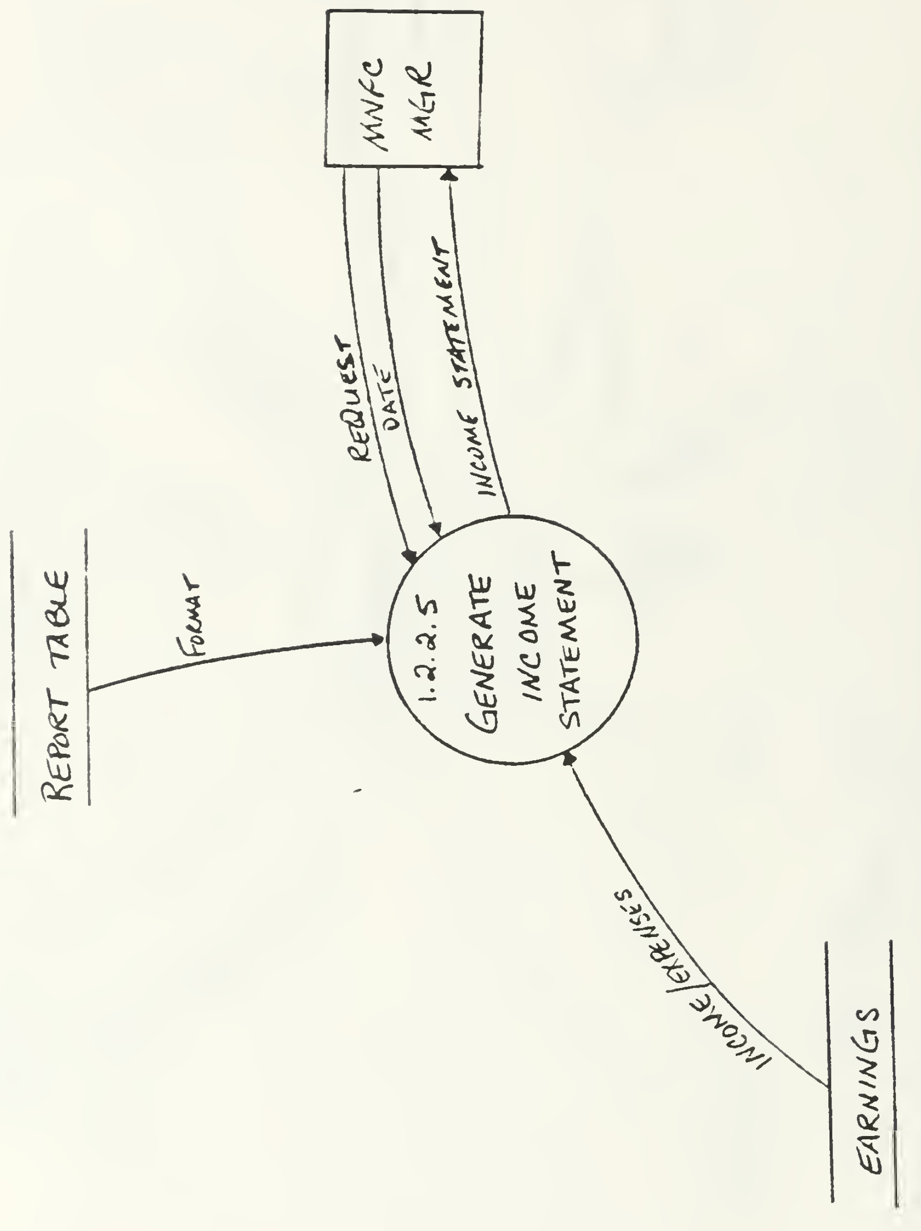


Figure A.21 Generation of Income Statements.

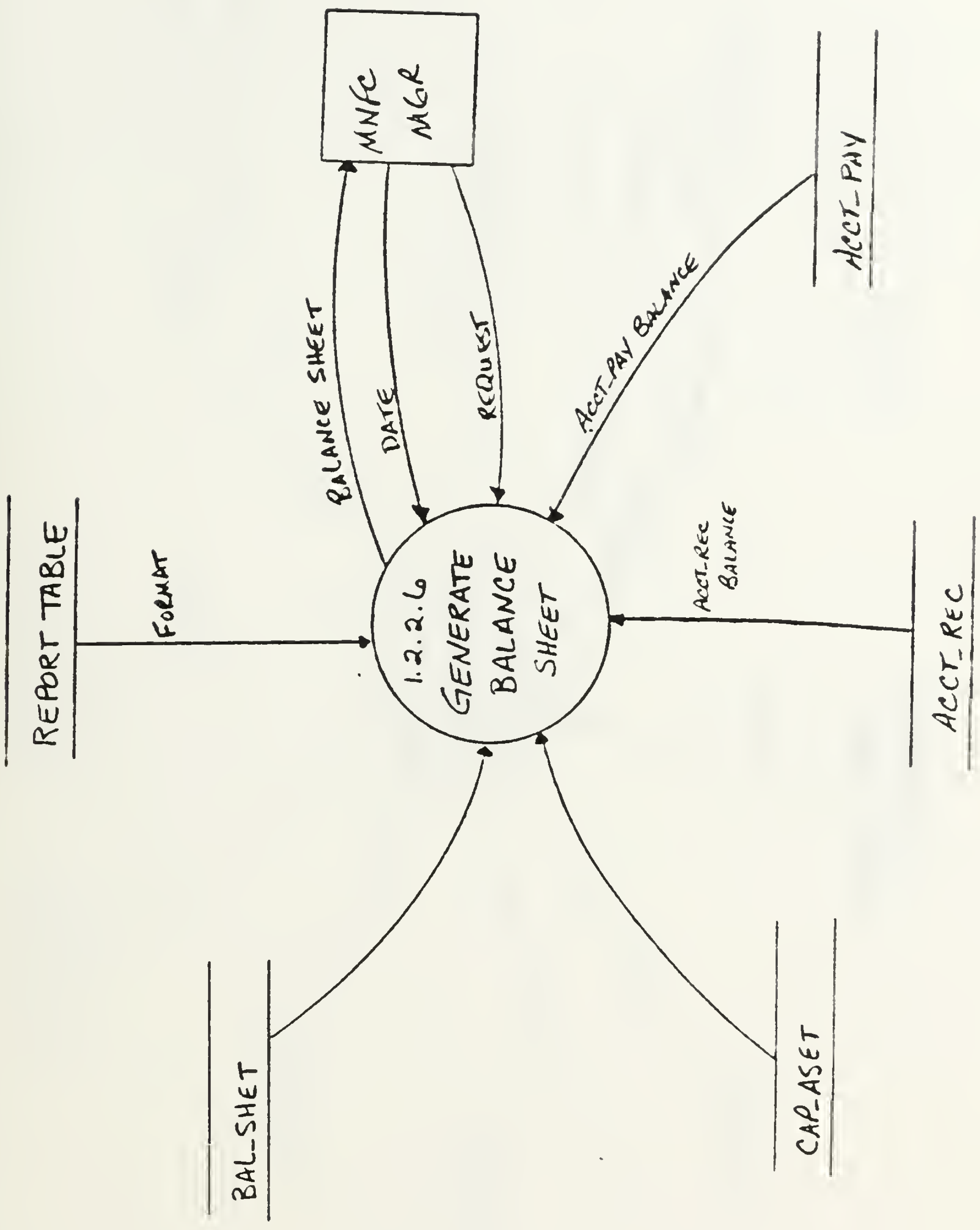


Figure A.22 Generation of Balance Sheets.

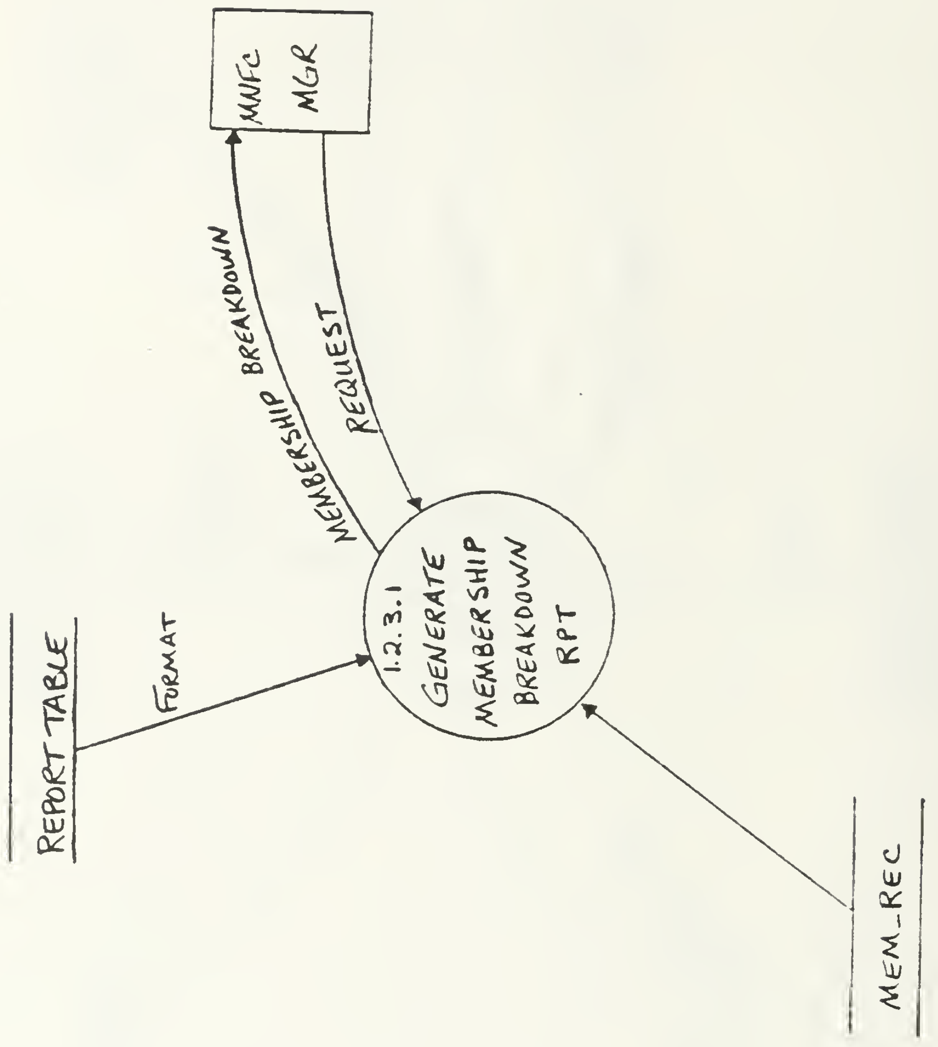


Figure A.23 Generation of Membership Breakdown Report.

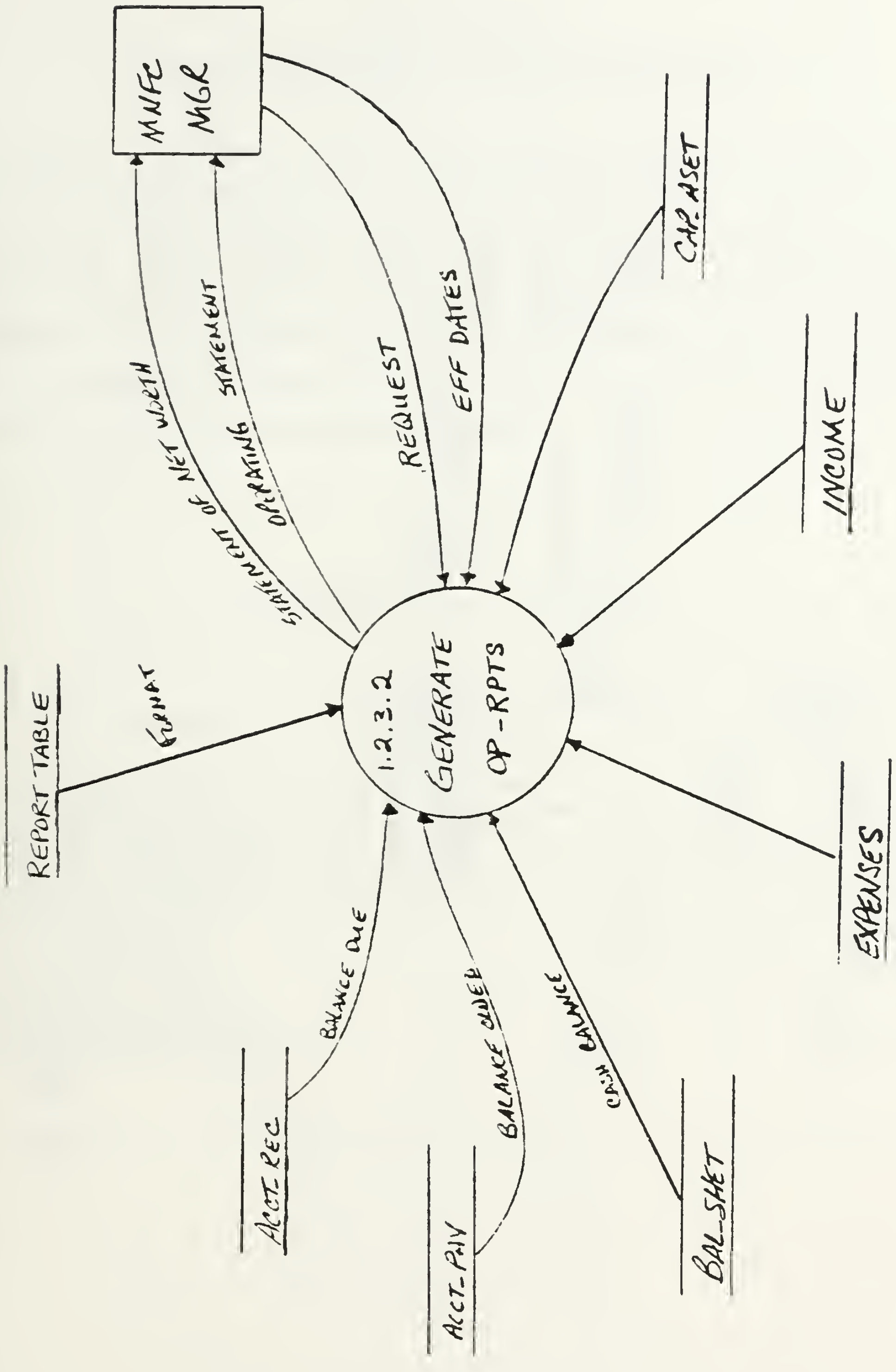


Figure A.24 Generation of Annual Operating Reports.

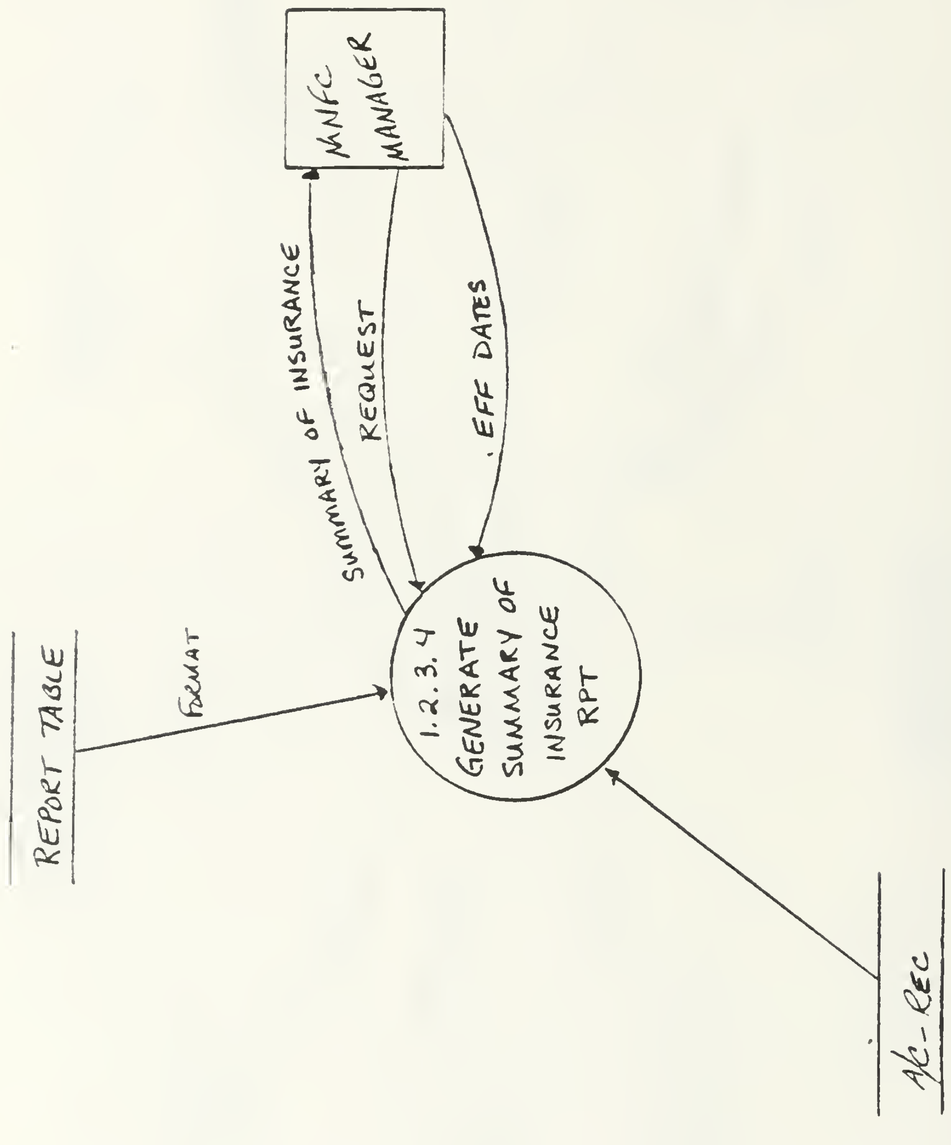


Figure A.25 Generation of the Annual Summary of Insurance Report.

APPENDIX B
SYSTEM REPORTS AND STATEMENTS

To assist the manager of the Monterey Navy Flying Club (MNFC) in the execution of his administrative duties, the MNFC Management Information System (MIS) produces various reports and statements. System operating procedures necessary to generate these documents are discussed in Chapter III, the User's Guide.

Examples of each document capable of being produced by the MNFC MIS prototype are displayed in Figure B.1 through Figure B.14. Tables 1 and 2 summarize the products by category and scheduled submission.

TABLE 1
REQUIREMENT CATEGORIES

Report/Statement	General Admin Report	Financial Report	OPNAV Annual Report
Membership Roster	Yes		
A/C Inventory & Status	Yes		
Monthly Manager's Rpt	Yes		
Aged Acct Receivables		Yes	
Instructor's Statement		Yes	
Member's Statement		Yes	
Lessor's Statement		Yes	
Income Statement		Yes	
Balance Sheet		Yes	
Membership Breakdown	Yes		Yes
Operating Statements		Yes	Yes
Statement of Net Worth		Yes	Yes
Summary of Insurance	Yes	Yes	Yes

TABLE 2

SUBMISSION SCHEDULES

Report/Statement	Monthly	Annually	As Needed
Membership Roster	Yes		Yes
A/C Inventory & Status	Yes		Yes
Monthly Manager's Rpt	Yes		
Aged Acct Receivables	Yes		Yes
Instructor's Statement	Yes		Yes
Member's Statement	Yes		Yes
Lessor's Statement	Yes		Yes
Income Statement		Yes	
Balance Sheet	Yes	Yes	Yes
Membership Breakdown		Yes	
Operating Statements		Yes	Yes
Statement of Net Worth		Yes	Yes
Summary of Insurance		Yes	Yes

```
                    MONTEREY NAVY FLYING CLUB ROSTER

  ROSTER DATE: 10/28/86      TOTAL MEMBERS: 237        PAGE: 13

NAME/ADDRESS               MEMBER    INSTRUCTOR    CATEGORY   STATUS
PHONE                       NO.         NO.
---------------------------------------------------------------------

Clift, Michael R.           1232                     AO       INACT
4538 Brownell
Monterey, CA 93940
(408)646-0732

George, Derek R.            2334       1345          NO
1284 Oakglen Way
Monterey, CA 93940
(408)649-0992

Graham, James M.            9999                     NO       INACT
27 Shubrick Road
Monterey, CA  93940
(408)373-7230
```

Figure B.1 Monterey Navy Flying Club Membership Roster.

```
                              MONTEREY NAVY FLYING CLUB

                  MONTHLY AIRCRAFT STATUS AND FLIGHT TIME REPORT

                             FOR PERIOD ENDING: 03/05/86

-----------------------------------------------------------------------------------------------------------------
A/C NUM -  TYPE  - ENDTIME -  START  - GROSS  - MAINHR - BILLED -  RENT  - RENT TOT - LEASE  - LEAS TOT - MAINT CST - INSUR COST
 OTHER  - NET TO OWNER - FUEL CST - NET TO CLUB -
-----------------------------------------------------------------------------------------------------------------
N15631  - P-180  - 140.00 - 100.00 - 40.00 - 5.030 - 35.00 - 30.00 - 1,050.00 - 15.00 - $525.00 -  $420.00 -   $60.00
        -          -          -          -

-----------------------------------------------------------------------------------------------------------------
N7322A  - C-150  - 100.00 - 70.000 - 30.00 - 0.000 - 30.00 - 23.00 - $690.00 - 11.00 - $330.00 -  $55.00 -   $60.00
        -          -          -          -

-----------------------------------------------------------------------------------------------------------------
N8847Y  - AZTEC  - 120.00 - 100.00 - 20.00 - 0.000 - 20.00 - 65.00 - 1,300.00 -       -   $0.00 -          -   $75.00
        -          -          -          -

-----------------------------------------------------------------------------------------------------------------
```

Figure B.2 Aircraft Inventory and Status Report.

```
                         MONTEREY NAVY FLYING CLUB
                             MANAGER'S REPORT

                             MONTH OF  JANUARY


DATE:   01/19/87                         MEMBERSHIP STATUS:


MONTHLY FLIGHT TIME:   146.000  HRS.      NEW MEMBERS            -0-

COLLECTIONS:       $    5,093.28          LOST MEMBERS           -0-

EXPENDITURES:      $    3,575.97          CURRENT TOTAL          200

EXPECTED MAJOR EXPENDITURES:


_________________________________________________________________

CURRENT ASSETS:                          CURRENT LIABILITIES:

  CASH-CHECKING                $ 1,234.50  ACCOUNTS PAYABLE $  2,345.67
  CASH-SAVINGS                  12,345.67  SHORT-TERM PAYABLE       0.00
  ACCOUNT RECEIVABLE-MBRS        1,234.56
  ACCOUNT RECEIVABLE-LESSOR          0.00
                               -----------                    -----------
        TOTAL               $  14,814.73       TOTAL      $  2,345.67
                               ___________                    ___________
                               -----------                    -----------
_________________________________________________________________

VIOLATIONS, INCIDENTS, OR ACCIDENTS:


COMMENTS:



                                          T. E. WOOLCOCK
                                          MANAGER
```

Figure B.3 MNFC Monthly Manager's Report.

```
                    MONTEREY NAVY FLYING CLUB
                    AGED ACCOUNTS RECEIVABLE

                    AS OF ____________________

  MEMBER/CLIENT             ACCT NO.   CURRENT  30 DAYS  60 DAYS  90 DAYS
-------------------------------------------------------------------------------

 George, Derek R.             2334     $ 56.50  $ 00.00  $ 00.00  $ 00.00
 1284 Oakglen Way
 Monterey, CA  93940

 Graham, James M.             9999     $ 00.00  $ 00.00  $ 00.00  $ 00.00
 27 Shubrick Rd
 Monterey, CA  93940

 Longfellow, David M.         3333     $ 60.00  $ 60.00  $ 00.00  $ 00.00
 123 Nowhere Lane
 Monterey, CA  93940
```

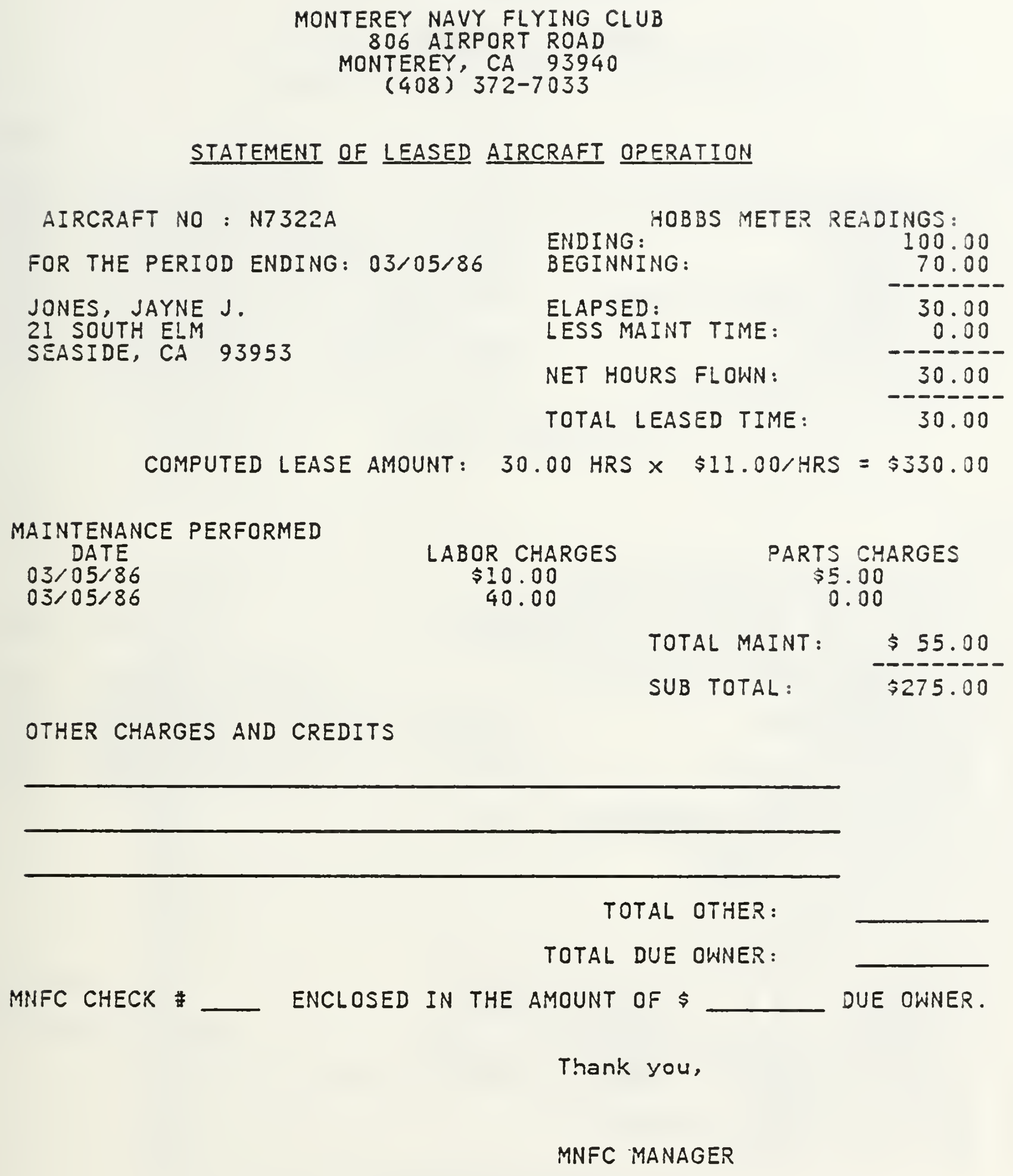
Figure B.4 MNFC Aged Accounts Receivables Report.

```
                 MONTEREY NAVY FLYING CLUB
                    806 AIRPORT ROAD
                  MONTEREY, CA  93940
                    (408) 372-7033

           STATEMENT OF INSTRUCTION PROVIDED

INSTRUCTOR NO.:  1234          STATEMENT FOR PERIOD ENDING: 10/28/86

 Derek George
 1284 Oakglen Way
 Monterey, CA  93940

  DATE          STUDENT           AIRCRAFT    HOURS FLOWN    AMOUNT
----------   ---------------     ----------  -------------  ---------
10/03/86     Clift                NB047Y         1.00         10.00

10/13/86     Rodeck               N7322A         1.50         15.00


                                   TOTAL DUE INSTRUCTOR:  $ 25.00
```

Figure B.5 MNFC Monthly Instructor's Statement.

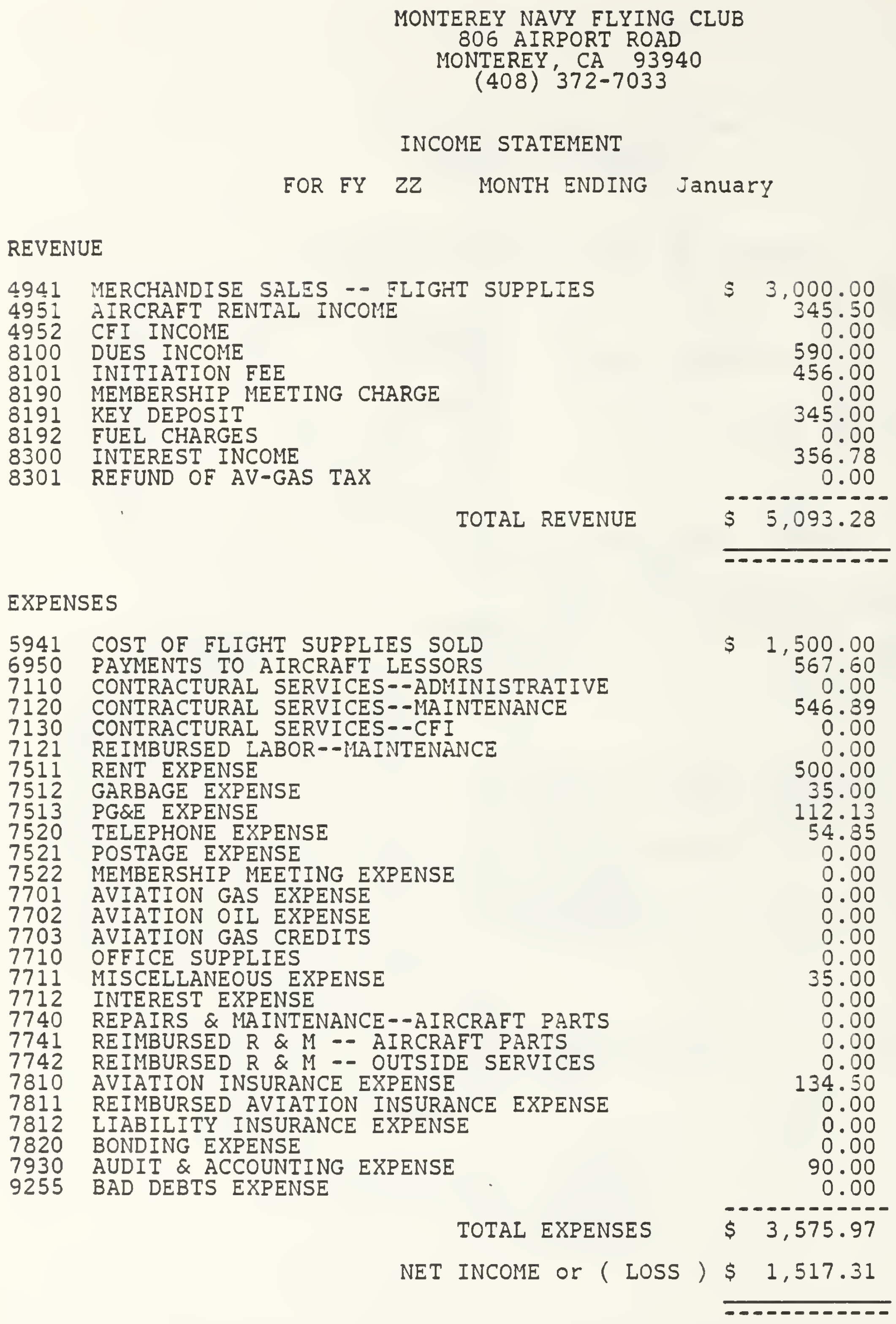
```
                    MONTEREY NAVY FLYING CLUB
                       806 AIRPORT ROAD
                     MONTEREY, CA  93940
                       (408) 372-7033


        GEORGE, DEREK R.                       ACCOUNT        DATE
        1284 OAKGLEN WAY                        2334        02/28/86
        MONTEREY, CA  93940



                    PAYMENT DUE BY 03/15/86


  DATE      INVOICE     DESCRIPTION          CHARGES     PAYMENTS   BALANCE
 -------   ---------   ------------------   ---------   ---------- ---------
                       BALANCE FORWARD                             $ 30.00
02/01/86     709       7HH/ 2.2 hrs         $  52.80
02/03/86     899       45B/ 1.3 hrs            45.50
02/03/86     899       GEORGE/GROUND           10.00
02/03/86     899       GEORGE/FLIGHT           15.00
02/13/86               PAYMENT                          $  85.00
02/28/86               DUES                    22.50
                                              -------   ---------   --------
                                            $ 175.80    $  85.00    $  90.80


      CURRENT     30 DAYS     60 DAYS   90 DAYS    AMOUNT DUE
      --------   ---------   --------- ---------  ------------
      $ 90.80    $ 00.00     $ 0.00    $ 0.00        $  90.80
```

Figure B.6 MNFC Monthly Member's Statement.



```
                    MONTEREY NAVY FLYING CLUB
                        806 AIRPORT ROAD
                      MONTEREY, CA  93940
                         (408) 372-7033

              STATEMENT OF LEASED AIRCRAFT OPERATION

  AIRCRAFT NO : N7322A                    HOBBS METER READINGS:
                                     ENDING:                100.00
 FOR THE PERIOD ENDING: 03/05/86     BEGINNING:              70.00
                                                          --------
 JONES, JAYNE J.                     ELAPSED:                30.00
 21 SOUTH ELM                        LESS MAINT TIME:         0.00
 SEASIDE, CA  93953                                       --------
                                     NET HOURS FLOWN:        30.00
                                                          --------
                                     TOTAL LEASED TIME:      30.00

       COMPUTED LEASE AMOUNT:  30.00 HRS x  $11.00/HRS = $330.00


MAINTENANCE PERFORMED
    DATE                   LABOR CHARGES          PARTS CHARGES
 03/05/86                     $10.00                 $5.00
 03/05/86                      40.00                  0.00

                                          TOTAL MAINT:    $ 55.00
                                                         ---------
                                          SUB TOTAL:      $275.00

 OTHER CHARGES AND CREDITS

 _______________________________________________________

 _______________________________________________________

 _______________________________________________________

                                     TOTAL OTHER:        _________

                                 TOTAL DUE OWNER:        _________

MNFC CHECK # ____  ENCLOSED IN THE AMOUNT OF $ ________ DUE OWNER.

                                  Thank you,


                                  MNFC MANAGER
```

Figure B.7 MNFC Monthly Lessor's Statement.



MONTEREY NAVY FLYING CLUB
806 AIRPORT ROAD
MONTEREY, CA 93940
(408) 372-7033

INCOME STATEMENT

FOR FY ZZ MONTH ENDING January

REVENUE

4941	MERCHANDISE SALES -- FLIGHT SUPPLIES	$ 3,000.00
4951	AIRCRAFT RENTAL INCOME	345.50
4952	CFI INCOME	0.00
8100	DUES INCOME	590.00
8101	INITIATION FEE	456.00
8190	MEMBERSHIP MEETING CHARGE	0.00
8191	KEY DEPOSIT	345.00
8192	FUEL CHARGES	0.00
8300	INTEREST INCOME	356.78
8301	REFUND OF AV-GAS TAX	0.00
	TOTAL REVENUE	$ 5,093.28

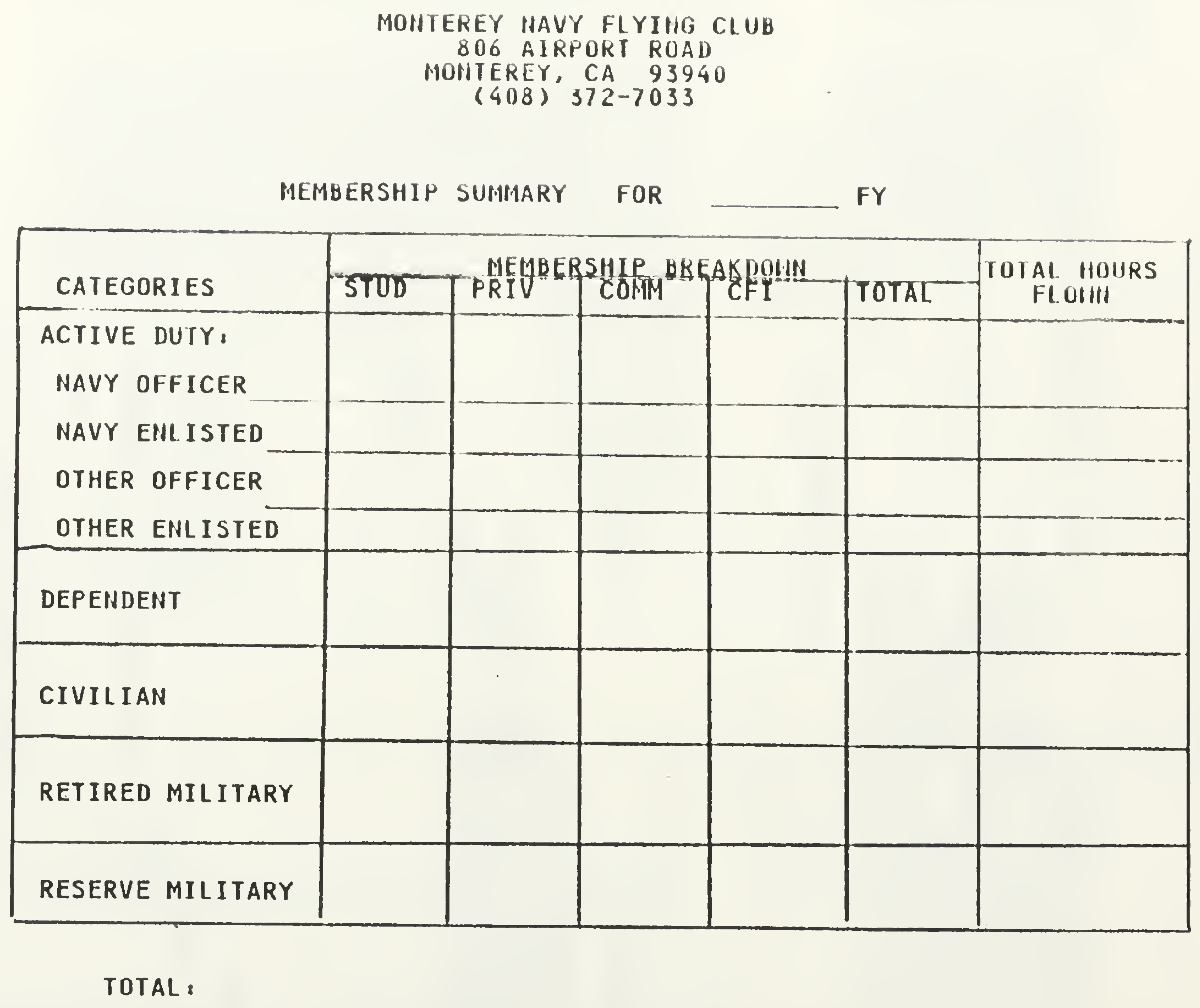
EXPENSES

5941	COST OF FLIGHT SUPPLIES SOLD	$ 1,500.00
6950	PAYMENTS TO AIRCRAFT LESSORS	567.60
7110	CONTRACTURAL SERVICES--ADMINISTRATIVE	0.00
7120	CONTRACTURAL SERVICES--MAINTENANCE	546.89
7130	CONTRACTURAL SERVICES--CFI	0.00
7121	REIMBURSED LABOR--MAINTENANCE	0.00
7511	RENT EXPENSE	500.00
7512	GARBAGE EXPENSE	35.00
7513	PG&E EXPENSE	112.13
7520	TELEPHONE EXPENSE	54.85
7521	POSTAGE EXPENSE	0.00
7522	MEMBERSHIP MEETING EXPENSE	0.00
7701	AVIATION GAS EXPENSE	0.00
7702	AVIATION OIL EXPENSE	0.00
7703	AVIATION GAS CREDITS	0.00
7710	OFFICE SUPPLIES	0.00
7711	MISCELLANEOUS EXPENSE	35.00
7712	INTEREST EXPENSE	0.00
7740	REPAIRS & MAINTENANCE--AIRCRAFT PARTS	0.00
7741	REIMBURSED R & M -- AIRCRAFT PARTS	0.00
7742	REIMBURSED R & M -- OUTSIDE SERVICES	0.00
7810	AVIATION INSURANCE EXPENSE	134.50
7811	REIMBURSED AVIATION INSURANCE EXPENSE	0.00
7812	LIABILITY INSURANCE EXPENSE	0.00
7820	BONDING EXPENSE	0.00
7930	AUDIT & ACCOUNTING EXPENSE	90.00
9255	BAD DEBTS EXPENSE	0.00
	TOTAL EXPENSES	$ 3,575.97
	NET INCOME or (LOSS)	$ 1,517.31

Figure B.8 MNFC Income Statement.

```
                         MONTEREY NAVY FLYING CLUB
                              806 AIRPORT ROAD
                           MONTEREY, CA  93940
                             (408) 372-7033

                    BALANCE SHEET AS OF  January 16, 19XX


ASSETS

1110    CASH -- CHECKING                                 $   1,234.50
1130    CASH -- SAVINGS                                     12,345.67
1210    ACCOUNTS RECEIVABLE -- MBRS                          1,234.56
1211    ACCOUNTS RECEIVABLE -- LESSORS                           0.00
1341    INVENTORY -- FLIGHT SUPPLIES                         2,045.76
1530    PREPAID EXPENSES -- INSURANCE                          456.50
1650    FIXED ASSETS -- AIRCRAFT                           154,000.00
1750    LESS ACCUMULATED DEPRECIATION                        2,345.50
1651    FIXED ASSETS -- SHOP EQUIPMENT                       2,345.00
1751    LESS ACCUMULATED DEPRECIATION                            0.00
                                                         ------------
                                       TOTAL ASSETS      $ 176,007.99
                                                         ____________
                                                         ------------



LIABILITIES

2100    ACCOUNTS PAYABLE                                    $2,345.67
2130    MEMBERS KEY DEPOSITS                                    45.00
2140    RESERVE FOR AIRCRAFT INSURANCE                         456.70
2510    SHORT TERM PAYABLES -- NOTES                             0.00
2530    SHORT TERM PAYABLES -- INSURANCE                         0.00
2710    UNEARNED DUES                                            0.00
                                                         ------------
                                   TOTAL LIABILITIES     $   2,847.37
                                                         ____________


NET WORTH

RETAINED EARNINGS                                          173,160.62
                                                         ------------
               TOTAL LIABILITIES AND NET WORTH           $ 176,007.99
                                                         ____________
                                                         ------------
```

Figure B.9 MNFC Balance Sheet.



MONTEREY NAVY FLYING CLUB
806 AIRPORT ROAD
MONTEREY, CA 93940
(408) 372-7033

MEMBERSHIP SUMMARY FOR ______ FY

CATEGORIES	MEMBERSHIP BREAKDOWN STUD	PRIV	COMM	CFI	TOTAL	TOTAL HOURS FLOWN
ACTIVE DUTY:						
NAVY OFFICER						
NAVY ENLISTED						
OTHER OFFICER						
OTHER ENLISTED						
DEPENDENT						
CIVILIAN						
RETIRED MILITARY						
RESERVE MILITARY						

TOTAL:

Figure B.10 MNFC Membership Breakdown Summary Report.

FLYING CLUB NAME:	MONTEREY NAVY FLYING CLUB
BALANCE SHEET AS OF:	September 30, 19XX
PREPARED BY:	T. E. Woolcock, Manager

Current		
104, 109	Cash	$ 13,960.12
121	Investments, short term	3,412.46
131, 132	Accounts Receivable	3,623.82
153	Inventories, resale	2,335.13
154	Inventories, aircraft parts	0.00
155	Inventories, fuel	0.00
	Total Current Assets	$ 33,328.53
Non Current		
173	Furniture, fixtures	$ 900.00
183	Less: Accum. Deprec.	900.00
171	Vehicles	0.00
181	Less: Accum. Deprec.	0.00
175	Bldgs. & Facilities	0.00
185	Less: Accum. Deprec.	0.00
178	Aircraft owned	60,123.60
188	Less: Accum. Deprec.	31,242.31
	Total Non Current Assets	$ 28,881.29
Other		
161	Prepaid expenses	$ 0.00
191	Long-term investments	0.00
195	Investments - funded reserves	0.00
	Total Other Assets	$ 0.00
Total Assets		$ 62,209.82

Figure B.11a Annual Operating Statement - Balance Sheet.

FLYING CLUB NAME: MONTEREY NAVY FLYING CLUB

BALANCE SHEET AS OF: September 30, 19XX

Current Liabilities		
201	Accounts payable	$ 1,374.30
211	Accrued wages	0.00
213, 215 214, 229	Employee benefits & taxes	0.00
212	Accrued annual leave	0.00
207	Insurance	0.00
261	Short term loans	0.00
	Total Current Liabilities	$ 1,374.30
Long Term		
272	Loans	0.00
	Total Liabilities	$ 1,374.30
Net Worth		
299	Funded reserves	$ 0.00
291	Retained earnings	60,835.52
	Total Net Worth	$ 60,835.52
	Total Liabilities and Net Worth	$ 62,209.82

Figure B.11b Annual Operating Statement - Balance Sheet (con't).

FLYING CLUB NAME: MONTEREY NAVY FLYING CLUB

OPERATING STATEMENT

FOR THE YEAR ENDED: September 30, 19XX

	SALES	
302	Flight supplies and accessories	$ 16,275.19
402	Less: Cost of flight supplies and accessories	14,834.29
	Gross Profit on Sales	$ 1,440.90
	SALE OF SERVICES NON-VA	
303	Aircraft rental	$ 199,702.31
305	In-flight instruction	0.00
307	Ground instruction	0.00
	Total Non-VA	$ 199,702.31
	SALE OF SERVICES VA	
304	Aircraft rental	$ 0.00
306	In-flight instruction	0.00
308	Ground instruction	0.00
	Total VA	$ 0.00
	Total Sale of Services	$ 199,702.31
	TOTAL SALES	$ 201,143.21
	DIRECT EXPENSE	
Personnel Costs		
603	Salaries and wages-admin	$ 19,129.92
604	Salaries and wages-maint	21,733.50
630	Social Security	0.00
631	Retirement annuity contributions	0.00
632	Group comprehensive medical contribution	0.00
633	Retirement life insurance contribution	0.00
622	Annual leave	0.00
623	Sick leave	0.00
	Total Personnel Cost	$ 40,863.42
Operating Costs		
Non-VA		
423	Aircraft rental	$ 42,803.95
427	Ground instruction	0.00
	Total Non-VA	$ 42,803.95
VA		
424	Aircraft rental	$ 0.00
428	Ground instruction	0.00
	Total VA	$ 0.00

Figure B.12a Annual Operating Statement - Income Statement.

FLYING CLUB NAME: MONTEREY NAVY FLYING CLUB

OPERATING STATEMENT

FOR THE YEAR ENDING: September 30, 19XX

	OTHER DIRECT	
421	Gasoline	$ 99,783.15
422	Oil	1,422.65
425	Storage and tiedown	0.00
426	Maintenance-routine	17,538.89
429	Maintenance-overhaul	0.00
641	Utilities and rent	6,759.82
661	Telephone and postage	2,002.73
721	Travel and per diem	0.00
731	Freight and transportation	0.00
701	Supplies	0.00
781	Insurance premiums	12,190.53
	Total Other Direct	$ 139,697.77
	Total Operating Costs and Other Direct	$ 182,501.72
	TOTAL OPERATING COSTS	
Depreciation		
762	Furniture, fixtures, eqpt	$ 0.00
761	Vehicles	0.00
763	Buildings and facilities	0.00
765	Aircraft	7,795.26
	Total Depreciation	$ 7,795.26
	Total Direct Expense	$ 231,160.40
Other Income		
531	Other dues and assessments	$ 45,388.25
569	Contributions and donations	0.00
	Interest Income	1,107.01
	Insurance proceeds	0.00
	Total Other Income	$ 46,495.26
Other Expense		
799	Miscellaneous other enpense	$ 12,963.94
905	Interest expense	15.70
904	Bad debt expense	53.15
	Total Other Expense	$ 13,032.79
999	NET INCOME (LOSS)	$ 3,445.28

Figure B.12b Annual Operating Statement - Income Statement (con't).

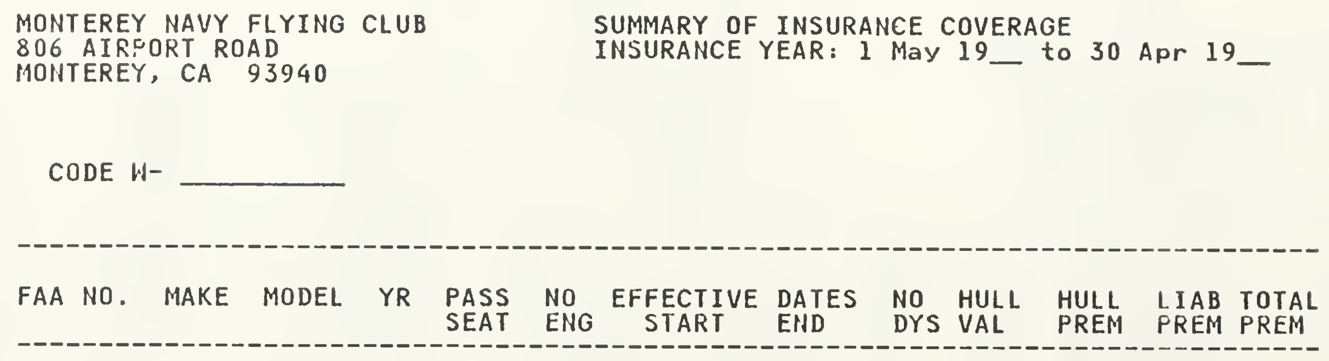
FLYING CLUB NAME: MONTEREY NAVY FLYING CLUB

STATEMENT OF NET WORTH

AS OF: September 30, 19XX

	299 Funded Reserves	291 Retained Earnings
Beginning of year balance	$ 0.00	$ 57,390.24
Add: Profit (999)	0.00	3,445.28
Deduct: Loss (999)		
Transfer of retained earnings to funded reserve	0.00	0.00
Transter of retained earnings from funded reserve	0.00	0.00
End of year balances	$ 0.00	$ 60,835.52

Figure B.13 Annual Statement of Net Worth.



```
MONTEREY NAVY FLYING CLUB           SUMMARY OF INSURANCE COVERAGE
806 AIRPORT ROAD                    INSURANCE YEAR: 1 May 19__ to 30 Apr 19__
MONTEREY, CA  93940

  CODE W- __________

-------------------------------------------------------------------------------

FAA NO.  MAKE  MODEL  YR  PASS  NO  EFFECTIVE DATES  NO  HULL  HULL  LIAB TOTAL
                          SEAT  ENG   START   END    DYS VAL   PREM  PREM PREM
-------------------------------------------------------------------------------
```

Figure B.14 Annual Summary of Insurance Coverage.

APPENDIX C
MNFC DATA DICTIONARY

Within the following pages the reader is provided various items to assist in his understanding of the physical structure of the Monterey Navy Flying Club (MNFC) Management Information System (MIS). The contents of this appendix are:

* System Hierarchy Charts (Figure C.1 through Figure C.3)
* A listing of all current Tables in the MNFC MIS Database
* A listing of all current Report formats in the MNFC MIS Database
* A listing of all current Entry/Edit Forms in the MNFC MIS
* Database table compositions
* A listing of Entry/Edit Forms and associated tables
* A listing of Reports and associated tables
* A listing of Database Attributes

The reader's attention is directed to the original prototype design specifications [Ref. 1] for definitions and table compositions listed but not contained in this appendix.

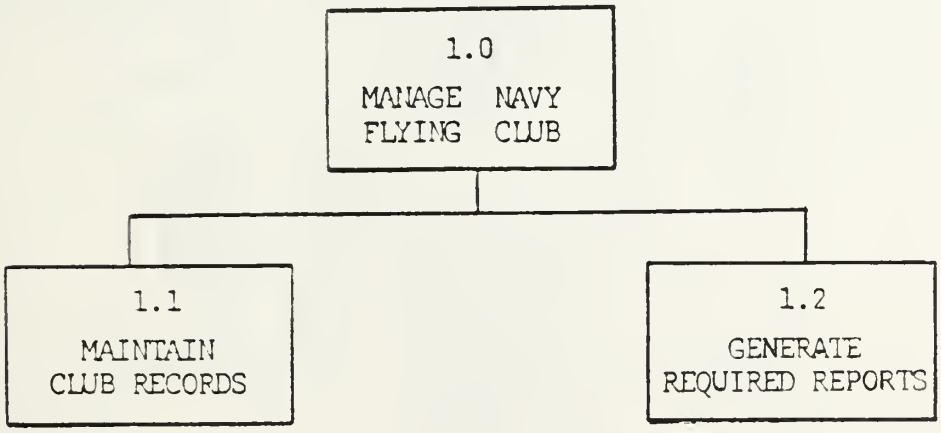


Figure C.1 MNFC System Hierarchy Chart.

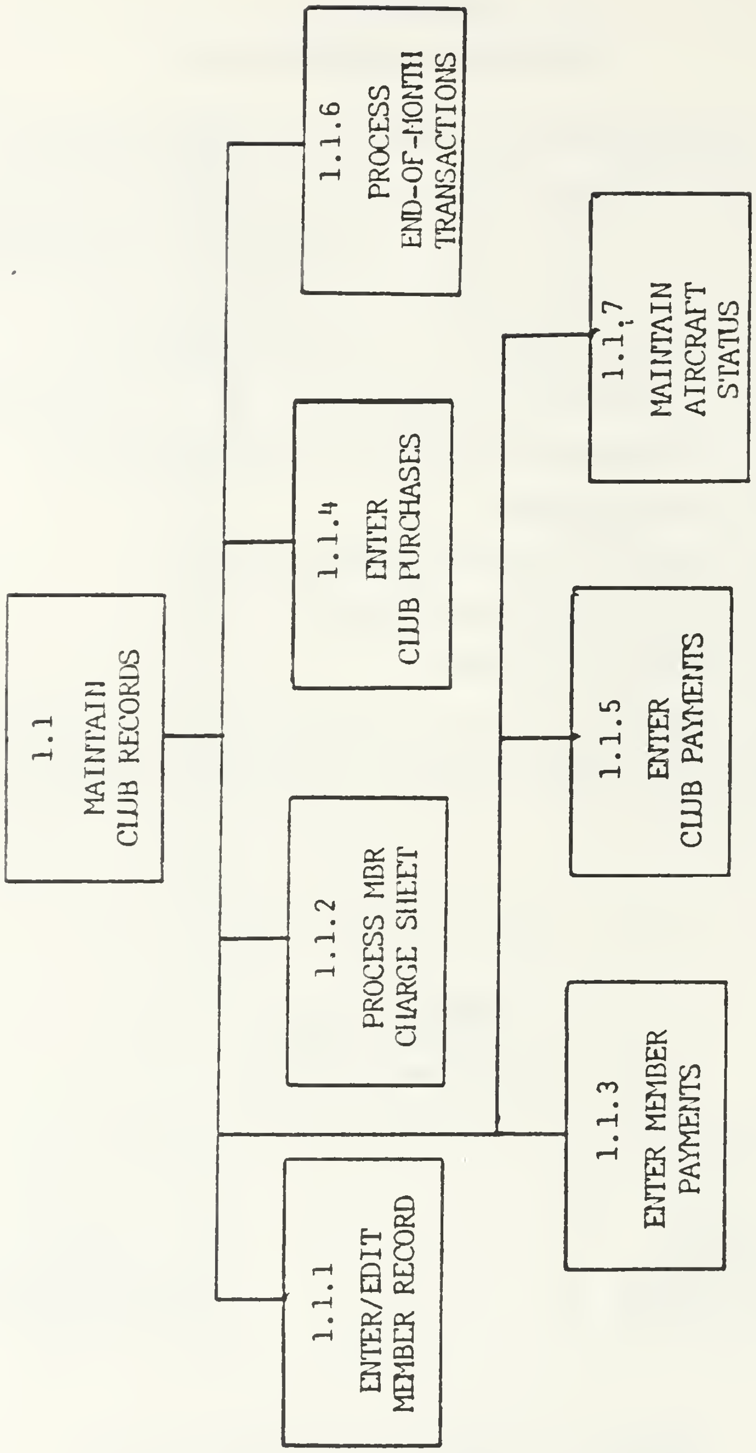


Figure C.2 MNFC System Hierarchy Chart (con't).

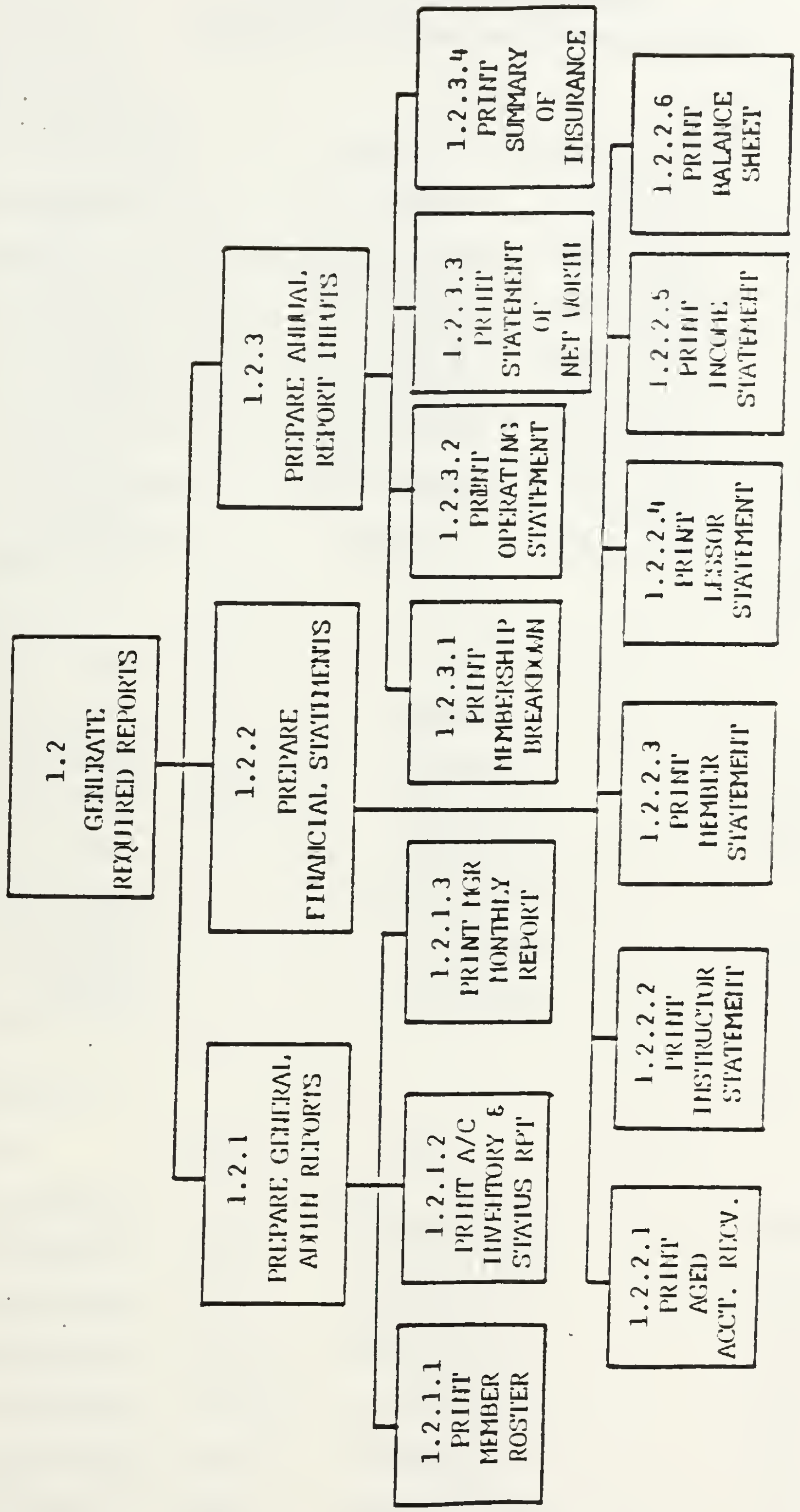

1.2 GENERATE REQUIRED REPORTS
1.2.1 PREPARE GENERAL ADMIN REPORTS
1.2.2 PREPARE FINANCIAL STATEMENTS
1.2.3 PREPARE ANNUAL REPORT INPUTS
1.2.1.1 PRINT MEMBER ROSTER
1.2.1.2 PRINT A/C INVENTORY & STATUS RPT
1.2.1.3 PRINT MGR MONTHLY REPORT
1.2.3.1 PRINT MEMBERSHIP BREAKDOWN
1.2.3.2 PRINT OPERATING STATEMENT
1.2.3.3 PRINT STATEMENT OF NET WORTH
1.2.3.4 PRINT SUMMARY OF INSURANCE
1.2.2.1 PRINT AGED ACCT. RECV.
1.2.2.2 PRINT INSTRUCTOR STATEMENT
1.2.2.3 PRINT MEMBER STATEMENT
1.2.2.4 PRINT LESSOR STATEMENT
1.2.2.5 PRINT INCOME STATEMENT
1.2.2.6 PRINT BALANCE SHEET

Figure C.3 MNFC System Hierarchy Chart (con't).

Monterey Navy Flying Club

Management Information System Database

Tables

A/C_HRS	A/CMAINT	A/C_REC	AP_PAID
ACCT_REC	AP_CHG	ACCT_PAY	APCGHIST
APPDHIST	AP_TEMP	BAL_JOUR	BAL_SHET
BILLTEMP	CAP_ASET	CHGHIST	EARNINGS
EXPENSE	EXP_HIST	FLT_HIST	FLT_REC
FORMS	HOURHIST	INC_HIST	INCOME
INST_REC	INSTHIST	JOURHIST	LES_ACCT
MAINHIST	MBR_SUM	MEM_CHG	MEM_FLT
MEM_PAY	MEM_REC	PAYHIST	REPORTS
TEMP_HRS			

Reports

A/C_STAT	AGED_AR	BALSHET1
BALSHET2	INCSTMT1	INCSTMT2
INSTSTMT	INS_SUM	LES_STMT
MAINT	MBR_BKDN	MEM_BILL
MGR_RPT	OPSRPT1	OPSRPT2
OPSRPT3	OPSRPT4	OPSRPT5
ROSTER		

Forms

A/C_FORM	ADDBALFM	ADDEARFM
ADJAPDFM	ADJCFIFM	ADJCHGFM
ADJEXPFM	ADJFLTHR	ADJINCFM
ADJMBRFL	ADJMBRFM	ADJMEMFM
ADJUSTFM	CHGFORM	CRED_PAY
CRED_PUR	FLTHRSFM	HOB_FORM
LES_FORM	MAINTFRM	MEM_FORM
PAYFORM		

Table: A/C_HRS

Read Password: NO

Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	A/C_NUM	TEXT	6 characters	Yes
2	CUM_HRS	REAL	1 value(s)	
3	HOB_END	REAL	1 value(s)	
4	HOBSTART	REAL	1 value(s)	
5	HOB_USED	REAL	1 value(s)	
6	HOBMAINT	REAL	1 value(s)	
7	HOB_NET	REAL	1 value(s)	
8	HOBOWNER	REAL	1 value(s)	
9	HOBLEASE	REAL	1 value(s)	
10	LEASEAMT	DOLLAR	1 value(s)	
11	TOTMAINT	DOLLAR	1 value(s)	

Table: A/CMAINT

Read Password: NO

Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	A/C_NUM	TEXT	6 characters	Yes
2	INVO_NUM	TEXT	4 characters	
3	LABOR_HR	REAL	1 value(s)	
4	LABR_CHG	DOLLAR	1 value(s)	
5	PART_CHG	DOLLAR	1 value(s)	
6	A/C_DATE	DATE	1 value(s)	
7	SOURCE	TEXT	50 characters	

Table: A/C_REC

Read Password: NO

Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	A/C_NUM	TEXT	6 characters	Yes
2	TYPE	TEXT	6 characters	
3	YEAR	TEXT	4 characters	
4	MODEL	TEXT	6 characters	
5	PASSEAT	TEXT	6 characters	
6	NO_ENG	INTEGER	1 value(s)	
7	HULLVAL	DOLLAR	1 value(s)	
8	HULLPREM	DOLLAR	1 value(s)	
9	LIABPREM	DOLLAR	1 value(s)	
10	TOTPREM	DOLLAR	1 value(s)	
11	INSSTART	DATE	1 value(s)	
12	INSSTOP	DATE	1 value(s)	
13	LEAS_RAT	DOLLAR	1 value(s)	
14	RENT_RAT	DOLLAR	1 value(s)	
15	PREP_INS	DOLLAR	1 value(s)	
16	BOOK_VAL	DOLLAR	1 value(s)	
17	ACC_DEP	DOLLAR	1 value(s)	
18	DEP_TERM	INTEGER	1 value(s)	
19	DATE_GOT	DATE	1 value(s)	

Table: AP_PAID
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	VENDNO	INTEGER	1 value(s)	Yes
2	PD:DATE	DATE	1 value(s)	
3	PD:AMT	DOLLAR	1 value(s)	
4	CHECK:NO	INTEGER	1 value(s)	

Table: ACCT_REC
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	WHO/WHAT	TEXT	20 characters	
2	AR_DATE	DATE	1 value(s)	
3	AMOUNT	DOLLAR	1 value(s)	
4	ACCT_NUM	TEXT	4 characters	Yes
5	MEM_NUM	TEXT	4 characters	Yes
6	LESOR_NU	TEXT	4 characters	Yes
7	CURR_BAL	DOLLAR	1 value(s)	
8	BAL:30	DOLLAR	1 value(s)	
9	BAL:60	DOLLAR	1 value(s)	
10	BAL:90	DOLLAR	1 value(s)	
11	BALFWD30	DOLLAR	1 value(s)	
12	BALFWD60	DOLLAR	1 value(s)	
13	BALFWD90	DOLLAR	1 value(s)	
14	SUM60+90	DOLLAR	1 value(s)	
15	BAL_FWD	DOLLAR	1 value(s)	

Table: AP_CHG
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	VENDNO	INTEGER	1 value(s)	Yes
2	PRODUCT	TEXT	20 characters	
3	P:DATE	DATE	1 VALUE(s)	
4	T:PRICE	DOLLAR	1 value(s)	
5	INV:NO	TEXT	15 characters	Yes

Table: ACCT_PAY
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	VENDNO	INTEGER	1 value(s)	Yes
2	VNL:NAME	TEXT	15 characters	
3	VNF:NAME	TEXT	15 characters	
4	VEN:MI	TEXT	2 characters	
5	VEN_STRT	TEXT	20 characters	
6	CITY	TEXT	15 characters	
7	STATE	TEXT	2 characters	
8	ZIPCODE	TEXT	5 charcaters	
9	PHONE	TEXT	13 characters	
10	CURRBAL	DOLLAR	1 value(s)	
11	POSTDATE	DATE	1 value(s)	

Table: APCGHIST
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	VENDNO	INTEGER	1 value(s)	Yes
2	PRODUCT	TEXT	20 characters	
3	P:DATE	DATE	1 VALUE(s)	
4	T:PRICE	DOLLAR	1 value(s)	
5	INV:NO	TEXT	15 characters	Yes

Table: APPDHIST
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	VENDNO	INTEGER	1 value(s)	Yes
2	PD:DATE	DATE	1 value(s)	
3	PD:AMT	DOLLAR	1 value(s)	
4	CHECK:NO	INTEGER	1 value(s)	

Table: AP_TEMP
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	VENDNO	INTEGER	1 value(s)	Yes
2	INV:NO	TEXT	15 characters	
3	P:DATE	DATE	1 value(s)	
4	T:PRICE	DOLLAR	1 value(s)	
5	PD:DATE	DATE	1 value(s)	
6	PD:AMT	DOLLAR	1 value(s)	

Table: BAL_JOUR
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	ACCT_NUM	TEXT	4 characters	Yes
2	ACCTNAME	TEXT	37 characters	
3	DOL:AMT	DOLLAR	1 value(s)	
4	TX:DATE	DATE	1 value(s)	
5	TX:CODE	TEXT	1 character	

Table: BAL_SHET
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	ACCT_NUM	TEXT	4 characters	Yes
2	ACCTNAME	TEXT	37 characters	
3	BALANCE	DOLLAR	1 value(s)	
4	BAL:DATE	DATE	1 value(s)	

Table: BILLTEMP
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	MEM_NUM	TEXT	4 characters	Yes
2	INVO_NUM	TEXT	4 characters	
3	TRANDATE	DATE	1 value(s)	
4	CHARGES	DOLLAR	1 value(s)	
5	PAYMENTS	DOLLAR	1 value(s)	
6	DESCRIPT	TEXT	20 characters	

Table: CAP_ASET
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	IDENT	TEXT	50 characters	Yes
2	ACCT_NUM	TEXT	4 characters	Yes
3	INIT_CST	DOLLAR	1 value(s)	
4	DATE_ACQ	DATE	1 value(s)	
5	DEP_TERM	INTEGER	1 value(s)	
6	ACC_DEP	DOLLAR	1 value(s)	

Table: CHGHIST
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	MEM_NUM	TEXT	4 characters	Yes
2	INVO_NUM	TEXT	4 characters	Yes
3	TRANDATE	DATE	1 value(s)	
4	CHARGES	DOLLAR	1 value(s)	
5	DESCRIPT	TEXT	20 characters	
6	HRS_FLON	REAL	1 value(s)	
7	A/C_NUM	TEXT	6 characters	

Table: EARNINGS
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	ACCT_NUM	TEXT	4 characters	Yes
2	ACCTNAME	TEXT	37 characters	
3	BALANCE	DOLLAR	1 value(s)	
4	BAL:DATE	DATE	1 value(s)	

Table: EXPENSE
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	ACCT_NUM	TEXT	4 characters	Yes
2	ACCTNAME	TEXT	37 characters	
3	EXP:AMT	DOLLAR	1 value(s)	
4	TX:DATE	DATE	1 value(s)	
5	TX:CODE	TEXT	1 character	

Table: EXP_HIST
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	ACCT_NUM	TEXT	4 characters	Yes
2	ACCTNAME	TEXT	37 characters	
3	EXP:AMT	DOLLAR	1 value(s)	
4	TX:DATE	DATE	1 value(s)	
5	TX:CODE	TEXT	1 character	

Table: FLT_HIST
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	FLTDATE	DATE	1 value(s)	
2	CATEGORY	TEXT	2 characters	Yes
3	FAA_CERT	TEXT	4 characters	Yes
4	FLT_HRS	REAL	1 value(s)	

Table: FLT_REC
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	FLTDATE	DATE	1 value(s)	
2	FLT_HRS	REAL	1 value(s)	
3	CATEGORY	TEXT	2 characters	Yes
4	FAA_CERT	TEXT	4 characters	Yes
5	A/C_NUM	TEXT	6 characters	

Table: HOURHIST
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	A/C_NUM	TEXT	6 characters	Yes
2	CUM_HRS	REAL	1 value(s)	
3	HOB_END	REAL	1 value(s)	
4	HOBSTART	REAL	1 value(s)	
5	HOB_USED	REAL	1 value(s)	
6	HOBMAINT	REAL	1 value(s)	
7	HOB_NET	REAL	1 value(s)	
8	HOBOWNER	REAL	1 value(s)	
9	HOBLEASE	REAL	1 value(s)	
10	LEASEAMT	DOLLAR	1 value(s)	

Table: INC_HIST
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	ACCT_NUM	TEXT	4 characters	Yes
2	ACCTNAME	TEXT	37 characters	
3	REV:AMT	DOLLAR	1 value(s)	
4	TX:DATE	DATE	1 value(s)	
5	TX:CODE	TEXT	1 character	

Table: INCOME
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	ACCT_NUM	TEXT	4 characters	Yes
2	ACCTNAME	TEXT	37 characters	
3	REV:AMT	DOLLAR	1 value(s)	
4	TX:DATE	DATE	1 value(s)	
5	TX:CODE	TEXT	1 character	

Table: INST_REC
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	INST_NUM	TEXT	4 characters	Yes
2	TRANDATE	DATE	1 value(s)	
3	MEM_NUM	TEXT	4 characters	Yes
4	HRS_FLON	REAL	1 value(s)	
5	A/C_NUM	TEXT	6 characters	
6	AMOUNT	DOLLAR	1 value(s)	
7	INVO_NUM	TEXT	4 characters	Yes

Table: INSTHIST
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	INST_NUM	TEXT	4 characters	Yes
2	TRANDATE	DATE	1 value(s)	
3	MEM_NUM	TEXT	4 characters	Yes
4	HRS_FLON	REAL	1 value(s)	
5	A/C_NUM	TEXT	6 characters	
6	AMOUNT	DOLLAR	1 value(s)	
7	INVO_NUM	TEXT	4 characters	Yes

Table: JOURHIST
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	ACCT_NUM	TEXT	4 characters	Yes
2	ACCTNAME	TEXT	37 characters	
3	DOL:AMT	DOLLAR	1 value(s)	
4	TX:DATE	DATE	1 value(s)	
5	TX:CODE	TEXT	1 character	

Table: LES_ACCT
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	LESOR_NU	TEXT	4 characters	Yes
2	L:NAME	TEXT	15 characters	
3	F:NAME	TEXT	15 characters	
4	MID_INIL	TEXT	2 characters	
5	SSN	TEXT	11 characters	
6	STREET	TEXT	20 characters	
7	CITY	TEXT	15 characters	
8	STATE	TEXT	2 characters	
9	ZIPCODE	TEXT	5 characters	
10	PHONE	TEXT	13 characters	
11	A/C_NUM	TEXT	6 characters	Yes

Table: MAINHIST

Read Password: NO

Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	A/C_NUM	TEXT	4 characters	Yes
2	INVO_NUM	TEXT	4 characters	Yes
3	LABOR_HR	REAL	1 value(s)	
4	LAB_CHG	DOLLAR	1 value(s)	
5	PART_CHG	DOLLAR	1 value(s)	
6	A/C_DATE	DATE	1 value(s)	
7	SOURCE	TEXT	50 characters	

Table: MBR_SUM

Read Password: NO

Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	SUMDATE	DATE	1 value(s)	
2	CATEGORY	TEXT	2 characters	Yes
3	CATNAME	TEXT	20 characters	
4	STUDCT	INTEGER	1 value(s)	
5	STUDHRS	REAL	1 value(s)	
6	PRIVCT	INTEGER	1 value(s)	
7	PRIVHRS	REAL	1 value(s)	
8	COMMCT	INTEGER	1 value(s)	
9	COMMHRS	REAL	1 value(s)	
10	CFICT	INTEGER	1 value(s)	
11	CFIHRS	REAL	1 value(s)	
12	TOTCOUNT	INTEGER	1 value(s)	
13	TOTHOURS	REAL	1 value(s)	

Table: MEM_CHG
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	MEM_NUM	TEXT	4 characters	Yes
2	INVO_NUM	TEXT	4 characters	Yes
3	TRANDATE	DATE	1 value(s)	
4	CHARGES	DOLLAR	1 value(s)	
5	DESCRIPT	TEXT	20 characters	
6	HRS_FLON	REAL	1 value(s)	
7	A/C_NUM	TEXT	6 characters	Yes

Table: MEM_FLT
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	MEM_NUM	TEXT	4 characters	Yes
2	CATEGORY	TEXT	2 characters	Yes
3	FAA_CERT	TEXT	4 characters	Yes
4	FLT_HRS	REAL	1 value(s)	
5	LAST_FLT	DATE	1 value(s)	
6	LAST_A/C	TEXT	6 characters	

Table: MEM_REC

Read Password: NO

Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	MEM_NUM	TEXT	4 characters	Yes
2	INST_NUM	TEXT	4 characters	Yes
3	L:NAME	TEXT	15 characters	
4	F:NAME	TEXT	15 characters	
5	MID_INIT	TEXT	2 characters	
6	SSN	TEXT	11 characters	
7	STREET	TEXT	20 characters	
8	CITY	TEXT	15 characters	
9	STATE	TEXT	2 characters	
10	ZIPCODE	TEXT	5 characters	
11	PHONE	TEXT	13 characters	
12	CATEGORY	TEXT	2 characters	Yes
13	INIT_FEE	TEXT	1 characters	
14	KEY_DEP	TEXT	1 characters	
15	MEM_STAT	TEXT	2 characters	Yes
16	DUE_STAT	TEXT	2 characters	
17	CUR_MODS	TEXT	30 characters	
18	COLLDUTY	TEXT	30 characters	
19	BIENNIAL	DATE	1 value(s)	
20	MED_CERT	DATE	1 value(s)	
21	MEDCLASS	TEXT	3 characters	
22	FAA_CERT	TEXT	4 characters	Yes
23	FAA_CLAS	TEXT	25 characters	
24	MISHAP	TEXT	1 characters	

Table: MEM_PAY
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	MEM_NUM	TEXT	4 characters	Yes
2	CHK_NUM	INTEGER	1 value(s)	
3	TRANDATE	DATE	1 value(s)	
4	PAYMENTS	DOLLAR	1 value(s)	
5	DESCRIPT	TEXT	20 characters	

Table: PAYHIST
Read Password: NO
Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	MEM_NUM	TEXT	4 characters	Yes
2	CHK_NUM	INTEGER	1 value(s)	
3	TRANDATE	DATE	1 value(s)	
4	PAYMENTS	DOLLAR	1 value(s)	
5	DESCRIPT	TEXT	20 characters	

Table: TEMP_HRS

Read Password: NO

Modify Password: NO

Column definitions

#	Name	Type	Length	Key
1	CUM_HRS	REAL	1 value(s)	
2	HOB_END	REAL	1 value(s)	
3	A/C_NUM	TEXT	6 characters	Yes

Application's Entry/Edit Form and Associated Table

Form Name	Associated Table
A/C_FORM	(VARIABLE FORM)
ADDBALFM	BAL_SHET
ADDEARFM	EARNINGS
ADJAPDFM	AP_PAID
ADJCFIFM	INST_REC
ADJCHGFM	AP_CHG
ADJEXPFM	EXPENSE
ADJFLTHR	FLT_REC
ADJINCFM	INCOME
ADJMBRFL	MEM_FLT
ADJMBRFM	MEM_CHG
ADJUSTFM	BAL_JOUR
CHGFORM	(VARIABLE FORM)
CRED_PAY	(VARIABLE FORM)
CRED_PUR	(VARIABLE FORM)
FLTHRSFM	A/C_HRS
HOB_FORM	(VARIABLE FORM)
LES_FORM	LES_ACCT
MAINTFRM	A/CMAINT
MEM_FORM	MEM_REC
PAYFORM	(VARIABLE FORM)

Application's Report Format and Associated Table

Report Name	Associated Table
A/C_STAT	A/C_HRS
AGED_AR	ACCT_REC
BALSHET1	BAL_SHET
BALSHET2	BAL_SHET
INCSTMT1	EARNINGS
INCSTMT2	EARNINGS
INSTSMT	INST_REC
INS_SUM	A/C_REC
LES_STMT	LES_ACCT
MAINT	A/CMAINT
MBR_BKDN	MBR_SUM
MEM_BILL	BILLTEMP
MGR_RPT	MEM_REC
OPSRPT1	TEMP_HRS
OPSRPT2	TEMP_HRS
OPSRPT3	TEMP_HRS
OPSRPT4	TEMP_HRS
OPSRPT5	TEMP_HRS
ROSTER	MEM_REC

Database Attributes and Table Locations

Name	Type	Length	Table	Key
A/C_DATE	DATE	1 value(s)	MAINHIST	
			A/CMAINT	
A/C_NUM	TEXT	6 characters	INST_REC	
			A/CMAINT	yes
			HOURHIST	yes
			MEM_CHG	
			LES_ACCT	yes
			FLT_REC	yes
			INSTHIST	
			A/C_REC	yes
			MAINHIST	yes
			TEMP_HRS	yes
			CHGHIST	
			A/C_HRS	yes
ACCTNAME	TEXT	37 characters	BAL_JOUR	
			INCOME	
ACCTNAME	TEXT	37 characters	JOURHIST	
			EXPENSE	
			INC_HIST	
			EXP_HIST	
			EARNINGS	
			BAL_SHET	
ACCT_NUM	TEXT	4 characters	EARNINGS	yes
			INCOME	yes
			BAL_SHET	yes
			CAP_ASET	yes
			JOURHIST	yes
			BAL_JOUR	yes
			EXP_HIST	yes
			EXPENSE	yes
			ACCT_REC	yes
			INC_HIST	yes

Name	Type	Length	Table	Key
ACC_DEP	DOLLAR	1 value(s)	CAP_ASET	
			A/C_REC	
AMOUNT	DOLLAR	1 value(s)	ACCT_REC	
			INSTHIST	
			INST_REC	
AR_DATE	DATE	1 value(s)	ACCT_REC	
BAL:30	DOLLAR	1 value(s)	ACCT_REC	
BAL:60	DOLLAR	1 value(s)	ACCT_REC	
BAL:90	DOLLAR	1 value(s)	ACCT_REC	
BAL:DATE	DATE	1 value(s)	BAL_SHET	
			EARNINGS	
BALANCE	DOLLAR	1 value(s)	BAL_SHET	
			EARNINGS	
BALFWD30	DOLLAR	1 value(s)	ACCT_REC	
BALFWD60	DOLLAR	1 value(s)	ACCT_REC	
BALFWD90	DOLLAR	1 value(s)	ACCT_REC	
BAL_FWD	DOLLAR	1 value(s)	ACCT_REC	
BIENNIAL	DATE	1 value(s)	MEM_REC	
BOOK_VAL	DOLLAR	1 value(s)	A/C_REC	
CATEGORY	TEXT	2 characters	FLT_HIST	yes
			MEM_FLT	yes
			MBR_SUM	yes
			MEM_REC	yes
			FLT_REC	yes
CATNAME	TEXT	20 characters	MBR_SUM	
CFICT	INTEGER	1 value(s)	MBR_SUM	
CFIHRS	REAL	1 value(s)	MBR_SUM	

Name	Type	Length	Table	Key
CHARGES	DOLLAR	1 value(s)	MEM_CHG	
			CHGHIST	
			BILLTEMP	
CHECK:NO	INTEGER	1 value(s)	AP_PAID	
			APPDHIST	
CHK_NUM	INTEGER	1 value(s)	MEM_PAY	
			PAYHIST	
CITY	TEXT	15 characters	LES_ACCT	
			MEM_REC	
			ACCT_PAY	
COLLDUTY	TEXT	30 characters	MEM_REC	
COMMCT	INTEGER	1 value(s)	MBR_SUM	
COMMHRS	REAL	1 value(s)	MBR_SUM	
CUM_HRS	REAL	1 value(s)	HOURHIST	
			TEMP_HRS	
			A/C_HRS	
CURRBAL	DOLLAR	1 value(s)	ACCT_PAY	
CURR_BAL	DOLLAR	1 value(s)	ACCT_REC	
CUR_MODS	TEXT	30 characters	MEM_REC	
DATE_ACQ	DATE	1 value(s)	CAP_ASET	
DATE_GOT	DATE	1 value(s)	A/C_REC	
DEP_TERM	INTEGER	1 value(s)	A/C_REC	
			CAP_ASET	
DESCRIPT	TEXT	20 characters	MEM_CHG	
			PAYHIST	
			CHGHIST	
			MEM_PAY	
			BILLTEMP	
DOL:AMT	DOLLAR	1 value(s)	BAL_JOUR	
			JOURHIST	
DUE_STAT	TEXT	2 characters	MEM_REC	
EXP:AMT	DOLLAR	1 value(s)	EXP_HIST	

Name	Type	Length	Table	Key
EXP:AMT	DOLLAR	1 value(s)	EXPENSE	
F:NAME	TEXT	15 characters	LES_ACCT	
			MEM_REC	
FAA_CERT	TEXT	4 characters	FLT_HIST	yes
			FLT_REC	yes
FAA_CERT	TEXT	4 characters	MEM_REC	yes
			MEM_FLT	yes
FAA_CLAS	TEXT	25 characters	MEM_REC	
FDATA	TEXT	80 characters	FORMS	
FLTDATE	DATE	1 value(s)	FLT_HIST	
			FLT_REC	
FLT_HRS	REAL	1 value(s)	FLT_HIST	
			MEM_FLT	
			FLT_REC	
FNAME	TEXT	8 characters	FORMS	yes
HOBLEASE	REAL	1 value(s)	HOURHIST	
			A/C_HRS	
HOBMAINT	REAL	1 value(s)	A/C_HRS	
			HOURHIST	
HOBOWNER	REAL	1 value(s)	HOURHIST	
			A/C_HRS	
HOBSTART	REAL	1 value(s)	A/C_HRS	
			HOURHIST	
HOB_END	REAL	1 value(s)	TEMP_HRS	
			A/C_HRS	
			HOURHIST	
HOB_NET	REAL	1 value(s)	HOURHIST	
			A/C_HRS	
HOB_USED	REAL	1 value(s)	A/C_HRS	
			HOURHIST	

Name	Type	Length	Table	Key
HRS_FLON	REAL	1 value(s)	CHGHIST	
			MEM_CHG	
			INST_REC	
			INSTHIST	
HULLPREM	DOLLAR	1 value(s)	A/C_REC	
HULLVAL	DOLLAR	1 value(s)	A/C_REC	
IDENT	TEXT	50 characters	CAP_ASET	yes
INIT_CST	DOLLAR	1 value(s)	CAP_ASET	
INIT_FEE	TEXT	1 characters	MEM_REC	
INSSTART	DATE	1 value(s)	A/C_REC	
INSSTOP	DATE	1 value(s)	A/C_REC	
INST_NUM	TEXT	4 characters	INST_REC	yes
			MEM_REC	
			INSTHIST	yes
INV:NO	TEXT	15 characters	AP_CHG	yes
			AP_TEMP	
			APCGHIST	yes
INVO_NUM	TEXT	4 characters	A/CMAINT	
			BILLTEMP	
			CHGHIST	yes
			INSTHIST	yes
			INST_REC	yes
			MEM_CHG	yes
INVO_NUM	TEXT	4 characters	MAINHIST	
KEY_DEP	TEXT	1 characters	MEM_REC	
L:NAME	TEXT	15 characters	MEM_REC	
			LES_ACCT	

Name	Type	Length	Table	Key
LABOR_HR	REAL	1 value(s)	A/CMAINT	
			MAINHIST	
LABR_CHG	DOLLAR	1 value(s)	MAINHIST	
			A/CMAINT	
LAST_A/C	TEXT	6 characters	MEM_FLT	
LAST_FLT	DATE	1 value(s)	MEM_FLT	
LEASEAMT	DOLLAR	1 value(s)	A/C_HRS	
			HOURHIST	
LEAS_RAT	DOLLAR	1 value(s)	A/C_REC	
LESOR_NU	TEXT	4 characters	LES_ACCT	yes
			ACCT_REC	yes
LIABPREM	DOLLAR	1 value(s)	A/C_REC	
MEDCLASS	TEXT	3 characters	MEM_REC	
MED_CERT	DATE	1 value(s)	MEM_REC	
MEM_NUM	TEXT	4 characters	CHGHIST	yes
			BILLTEMP	yes
			INST_REC	yes
			MEM_PAY	yes
			INSTHIST	yes
			PAYHIST	yes
			MEM_REC	yes
			ACCT_REC	yes
			MEM_FLT	yes
			MEM_CHG	yes
MEM_STAT	TEXT	1 characters	MEM_REC	yes
MID_INIT	TEXT	2 characters	MEM_REC	
			LES_ACCT	
MISHAP	TEXT	1 characters	MEM_REC	
MODEL	TEXT	6 characters	A/C_REC	
NO_ENG	INTEGER	1 value(s)	A/C_REC	

Name	Type	Length	Table	Key
P:DATE	DATE	1 value(s)	AP_CHG	
			APCGHIST	
			AP_TEMP	
PART_CHG	DOLLAR	1 value(s)	MAINHIST	
			A/CMAINT	
PASSEAT	TEXT	6 characters	A/C_REC	
PAYMENTS	DOLLAR	1 value(s)	MEM_PAY	
			BILLTEMP	
			PAYHIST	
PD:AMT	DOLLAR	1 value(s)	APPDHIST	
			AP_PAID	
			AP_TEMP	
PD:DATE	DATE	1 value(s)	APPDHIST	
			AP_TEMP	
PD:DATE	DATE	1 value(s)	AP_PAID	
PHONE	TEXT	13 characters	MEM_REC	
			ACCT_PAY	
			LES_ACCT	
POSTDATE	DATE	1 value(s)	ACCT_PAY	
PREP_INS	DOLLAR	1 value(s)	A/C_REC	
PRIVCT	INTEGER	1 value(s)	MBR_SUM	
PRIVHRS	REAL	1 value(s)	MBR_SUM	
PRODUCT	TEXT	20 characters	AP_CHG	
			APCGHIST	
RDATA	TEXT	132 characters	REPORTS	
RENT_RAT	DOLLAR	1 value(s)	A/C_REC	
REV:AMT	DOLLAR	1 value(s)	INCOME	
			INC_HIST	

Name	Type	Length	Table	Key
RNAME	TEXT	8 characters	REPORTS	yes
SOURCE	TEXT	50 characters	MAINHIST	
SOURCE	TEXT	50 characters	A/CMAINT	
SSN	TEXT	11 characters	LES_ACCT	
			MEM_REC	
STATE	TEXT	2 characters	LES_ACCT	
			MEM_REC	
			ACCT_PAY	
STREET	TEXT	20 characters	LES_ACCT	
			MEM_REC	
STUDCT	INTEGER	1 value(s)	MBR_SUM	
STUDHRS	REAL	1 value(s)	MBR_SUM	
SUM60+90	DOLLAR	1 value(s)	ACCT_REC	
SUMDATE	DATE	1 value(s)	MBR_SUM	
T:PRICE	DOLLAR	1 value(s)	AP_CHG	
			APCGHIST	
			AP_TEMP	
TOTCOUNT	INTEGER	1 value(s)	MBR_SUM	
TOTHOURS	REAL	1 value(s)	MBR_SUM	
TOTMAINT	DOLLAR	1 value(s)	A/C_HRS	
TOTPREM	DOLLAR	1 value(s)	A/C_REC	
TRANDATE	DATE	1 value(s)	CHGHIST	
			MEM_PAY	
			MEM_CHG	
			INSTHIST	
			INST_REC	
			BILLTEMP	
			PAYHIST	

Name	Type	Length	Table	Key
TX:CODE	TEXT	1 characters	JOURHIST	
			BAL_JOUR	
			EXP_HIST	
			INC_HIST	
			INCOME	
			EXPENSE	
TX:DATE	DATE	1 value(s)	INCOME	
			EXP_HIST	
			JOURHIST	
			INC_HIST	
			EXPENSE	
			BAL_JOUR	
TYPE	TEXT	6 characters	A/C_REC	
VEN:MI	TEXT	2 characters	ACCT_PAY	
VENDNO	INTEGER	1 value(s)	AP_TEMP	yes
			APCGHIST	yes
			AP_PAID	yes
			ACCT_PAY	yes
			APPDHIST	yes
			AP_CHG	yes
VEN_STRT	TEXT	20 characters	ACCT_PAY	
VNF:NAME	TEXT	15 characters	ACCT_PAY	
VNL:NAME	TEXT	15 characters	ACCT_PAY	
WHO/WHAT	TEXT	20 characters	ACCT_REC	
YEAR	TEXT	4 characters	A/C_REC	
ZIPCODE	TEXT	5 characters	MEM_REC	
			LES_ACCT	
			ACCT_PAY	

APPENDIX D
MODULE FUNCTIONAL DESCRIPTIONS

The Monterey Navy Flying Club MIS Prototype design calls for one central relational database, named FLYCLUB, and five application programs which access FLYCLUB. Those applications are:

- FLYSTART.APP -- Flying Club Startup Application
- CLUB_MIS.APP -- Club's Membership Information Application
- CLUB_FIN.APP -- Club's Financial Application
- CLUB_PRT.APP -- Club's Printing Application
- CLUB_END.APP -- Club's End-of-the-Month Processing Application.

Each application consists of program modules called Command Files within the R:BASE 5000 language. Functional descriptions of each module (Command File), grouped by their associated application, are contained in this appendix.

1. FLYSTART.APP

```
************************************************************************

Module:         CLUBMIS.CMD

Menu Option:    MEMBERSHIP TRANSACTIONS

Description:    This module calls the first R:BASE Application program
                to use the FLYCLUB Database of the MNFC MIS Prototype
                system.  The Membership Transactions menu is called
                allowing the user to make entries, deletions, and
                modifications to membership records.

Tables used:    NONE
Forms used:     NONE
************************************************************************

Module:         CLUBFIN.CMD

Menu Option:    FINANCIAL TRANSACTIONS

Description:    This module calls the Financial Application program of the
                MNFC MIS Prototype system.  The Financial Transactions
                menu is called allowing the user to choose which typical
                club financial transaction he desires to handle.

Tables used:    NONE
Forms used:     NONE
************************************************************************
```

Module: FLYMAINT.CMD

Menu Option: AIRCRAFT MAINTENANCE/FLIGHT HOUR UPDATES

Description: This module allows the user the choice of entering aircraft maintenance transactions or update aircraft flight hours for a specific aircraft. The user is shown a menu upon entry to the module and must provide the aircraft's tail number to access the database.

Tables used: A/C_HRS, A/CMAINT
Forms used: MAINTFRM, FLTHRSFM

****

Module: AC_UPDAT.CMD

Menu Option: AIRCRAFT INVENTORY UPDATE

Description: This module is used by the club manager to either add the necessary data on each individual aircraft placed in the MNFC inventory whether by lease or purchase. The manager is also able to delete an aircraft from the club's current inventory.

Tables used: A/C_HRS, CAP_ASET, LES_ACT, A/C_REC
Forms used: A/C_FORM, HOB_FORM, LES_FORM

****

Module: CLUBPRT.CMD

Menu Option: PRINT REPORTS/STATEMENTS

Description: This module calls the club's Printing Application Menu. From this menu the manager can choose which typical report or statement he desires printed. The report formats are located within the Reports Table of the FLYCLUB database.

Tables used: NONE
Forms used: NONE

****

Module: CLUBEOMT.CMD

Menu Option: END-OF-THE-MONTH TRANSACTIONS

Description: This module calls the End-of-the-Month Application Menu for the MNFC MIS system using the FLYCLUB database. The manager decides which function is required. Then from within this application all monthly bills are processed and accounts are posted and updated. Historical data is transferred to its appropriate database table during the execution of the individual processes.

Tables used: NONE
Forms used: NONE

****

2. CLUB_MIS.APP

****

Module: ADD_MEM.CMD

Menu Option: ENTER A NEW MEMBER RECORD

Description: This module utilizes a predesigned screen input form to enter new member personal data to the database. The form is recalled after each entry allowing the manager to make multiple additions to the membership rolls.

Tables used: MEM_REC
Forms used: MEM_FORM

****

Module: EDIT_MEM.CMD

Menu Option: EDIT/DELETE AN EXISTING MEMBER RECORD

Description: This module uses the same screen entry format as in adding a new member, but allows the user to change any data currently held on a member. The user is given the last member data and can then either edit or delete the record. The member's last name or member number are used to retrieve the database record.

Tables used: MEM_REC
Forms used: MEM_FORM

****

3. CLUB_FIN.APP

****

Module: PUTCHG.CMD

Menu Option: ENTER A MEMBER'S CHARGES

Description: This module uses a R:BASE variable form to take individual charge sheet data from a member and post members rental cost, flight hours, instruction received, aircraft used, etc.. The member's cumulative flight hours and the plane's hours are updated. All income earned and expenses paid are posted to the appropriate tables.

Tables used: MEM_REC, MEM_CHG, MEM_FLT, INST_REC, INCOME, EXPENSE, A/C_REC, A/C_HRS, FLT_REC
Forms used: CHGFORM

****

Module:	PUTPAY.CMD
Menu Option:	ENTER A MEMBER'S PAYMENTS
Description:	This module uses a R:BASE variable form to enter receipt of a members payment on account. The member's last name or member number is required to initialize the routine.
Tables used:	MEM_REC, MEM_PAY, BAL_JOUR
Forms used:	PAYFORM

****

Module:	CRED_BUY.CMD
Menu Option:	POST AN ACCOUNTS PAYABLE PURCHASE/INTEREST
Description:	This module allows the manager to enter all purchases made on credit to either a new supplier/vendor or to an existing account. The manager must provide the correct account number for inventory/asset/expense account increases. All interest charged to an account must be entered labeling attribute "PRODUCT" as interest and attritube "INVOICE NUMBER" as zero.
Tables used:	BAL_JOUR, EXPENSE, ACCT_PAY, AP_CHG
Forms used:	CRED_PUR

****

Module:	CASH_BUY.CMD
Menu Option:	POST CASH PURCHASE OF SERVICES/GOODS
Description:	This module provides the manager with a method to post check(cash) purchases made by the club. The transaction decreases the cash account and increases the inventory and expense accounts effected in the purchase. The manager must enter the correct account numbers to be credited/debted.
Tables used:	BAL_JOUR, EXPENSE
Forms used:	NONE

****

Module:	PD_ACCT.CMD
Menu Option:	POST PAYMENT TO AN ACCOUNTS PAYABLE ACCOUNT
Description:	This module allows the manager to enter payments made on accounts payable accounts. The manager must provide the creditor's account number or name, plus the payment data. A RBASE variable form is used for screen entry.
Tables used:	BAL_JOUR, AP_PAID, ACCT_PAY
Forms used:	CRED_PAY

****

Module:	EDIT_CHG.CMD
Menu Option:	EDIT MEMBER CHARGE TRANSACTION
Description:	This module allows editing or deleting of a member's charge to his account that may have been entered in error. The manager must know the member number, date of charge, invoice/charge sheet number to enter the change. The manager must indicate whether the error was discovered prior to or after the End-of-the-Month billing. The module handles the transaction differently based upon the time of error discovery/corrective entry.
Tables used:	MEM_REC, MEM_CHG, MEM_FLT, INST_REC, A/C_REC, INCOME, A/C_HRS, FLT_REC
Forms used:	ADJMBRFM, ADJCFIFM, ADJINCFM, ADJEXPFM, FLTHRSFM ADJMBRFL, ADJFLTHR

****

Module:	EDIT_PAY.CMD
Menu Option:	EDIT MEMBER PAYMENT TRANSACTION
Description:	This module allows editing or deleting a member's payment to his account if in error. The manager must indicate whether the error was discovered prior to or after the End-of-the-Month billing. The module handles the transaction differently based upon the time of error discovery/corrective entry. The manager must know the member's number and the original payment amount made and the amount of adjustment.
Tables used:	MEM_REC, MEM_PAY, BAL_JOUR
Forms used:	ADJMEMFM, ADJUSTFM

****

Module:	EDIT_AP.CMD
Menu Option:	EDIT ACCOUNTS PAYABLE PURCHASE
Description:	This module allows editing or deleting of purchases made to an accounts payable account. The manager provides the account numbers of the expense and asset accounts affected by the change, plus the amount of the adjustment to each account.
Tables used:	ACCT_PAY, BAL_JOUR, AP_CHG, EXPENSE
Forms used:	ADJCHGFM, ADJUSTFM, ADJEXPFM

****

Module:	EDIT_PD.CMD
Menu Option:	EDIT PAYMENT TO AN ACCOUNTS PAYABLE ACCOUNT
Description:	This module allows editing or deleting of payments made to an accounts payable account. The manager must indicate whether the error was discovered prior to or after the End-of-the-Month transactions have been processed. Corrections to the accounts are handled differently based upon when the error was discovered and when the corrective entry was made. The manager must know the accounts payable account number and payment data concerning the correction.

Tables used: ACCT_PAY, AP_PAID, BAL_JOUR, AP_CHG
Forms used: ADJAPDFM, ADJUSTFM

****

Module: REV_EDIT.CMD

Menu Option: REVENUE/SALE ACCOUNT ENTRY

Description: This module allows the manager to make compensating entries to any revenue account. The manager must know the revenue account number to make the corrective entry. The user will be asked if the entry is a plus or minus transaction and the amount.

Tables used: EARNINGS, INCOME
Forms used: ADJINCFM, ADDEARFM

****

Module: EXP_EDIT.CMD

Menu Option: EXPENSE ACCOUNT ENTRY

Description: This module allows the manager to make compensating entries to any expense account. The manager must know the expense account number to make the corrective entry. The user will be asked if the entry is a plus or minus transaction and the amount.

Tables used: EARNINGS, EXPENSE
Forms used: ADJEXPFM, ADDEARFM

****

Module: ASSTEDIT.CMD

Menu Option: ASSET ACCOUNT ENTRY

Description: This module allows the manager to make compensating entries to any asset account. The manager must know the asset account number to make the corrective entry. The module will ask if the entry is a plus or minus transaction and the amount.

Tables used: BAL_SHET, BAL_JOUR
Forms used: ADJUSTFM, ADDBALFM

****

Module: LIABEDIT.CMD

Menu Option: LIABILITY ACCOUNT ENTRY

Description: This module allows the manager to make compensating entries to any liability account. The manager must know the account number to make the corrective entry. The module will ask if the entry is a plus or minus transaction and the amount.

Tables used: BAL_SHET, BAL_JOUR
Forms used: ADJUSTFM, ADDBALFM

****

4. CLUB_PRT.APP

```
***********************************************************************

Module:         ROSTER.CMD

Menu Option:    PRINT MEMBER ROSTER

Description:    This module prepares the system printer and gathers the
                necessary data from the membership table to print the
                Monterey Navy Flying Club's Membership Roster which will
                include both the active and inactive membership.

Table used:     MEM_REC
Forms used:     NONE
Report used:    CLUB ROSTER

***********************************************************************

Module:         MGR_PRT.CMD

Menu Option:    PRINT MANAGER'S MONTHLY REPORT

Description:    This module prepares the monthly Manager's Report.
                The report must be completed after the monthly billing
                processes have been accomplished, otherwise, the data
                will be inaccurate (at a minimum, a month old).

Tables used:    BAL_SHET, EARNINGS, FLT_HIST, MEM_REC
Forms used:     NONE
Report used:    MGR_RPT

***********************************************************************

Module:         ACSTAT.CMD

Menu Option:    PRINT A/C INVENTORY & STATUS REPORT

Description:    This module prints the Aircraft Inventory and Status
                Report.  The module also updates the A/C_HRS table and
                readies it for the next month's entries.

Tables used:    A/C_HRS
Forms used:     FLTHRSFM
Report used:    A/C_STAT

***********************************************************************

Module:         PRTMEMBD.CMD

Menu Option:    PRINT MEMBERSHIP BREAKDOWN

Description:    This module prepares the Membership Breakdown Report
                showing which category of member, such as Naval Officer,
                Army Officer, has flown for how many hours over the past
                year per FAA certification.

Tables used:    MBR_SUM
Forms used:     NONE
Report used:    MBR_BKDN

***********************************************************************
```

```
Module:         OP_STMT.CMD

Menu Option:    PRINT ANNUAL OPERATING STATEMENT INPUTS

Description:    This module produces rough inputs for the required
                operating statements including statement of net worth
                for the annual report input in accordance with  CNO
                INST.  This report must be printed after the monthly
                postings in order for the data to be accurate.

                TEMP_HRS is listed as the associated table for this
                report.  This relationship allows the use of the
                report formatter to use data computed in this module
                in the report.

Tables used:    BAL_SHET, EARNINGS
Forms used:     NONE
Report used:    OPSRPT1, OPSRPT2, OPSRPT3, OPSRPT4, OPSRPT5

************************************************************************

Module:         INS_SUM.CMD

Menu Option:    PRINT SUMMARY OF INSURANCE

Description:    This module prepares the Summary of Insurance on all
                aircraft in the current inventory.  This report is
                one of the required annual report inputs.

Tables used:    A/C_REC
Forms used:     NONE
Report used:    INS_SUM

************************************************************************

Module:         AGED_AR.CMD

Menu Option:    PRINT AGED ACCOUNTS RECEIVABLE

Description:    This module prepares and prints the Summary of Aged
                accounts receivable accounts and their balances.

Tables used:    MEM_REC, ACCT_REC
Forms used:     NONE
Report used:    AGED_AR

************************************************************************

Module:         PRT_BAL.CMD

Menu Option:    PRINT BALANCE SHEET

Description:    This module takes the information located in the Balance
                Sheet table and prepares and prints a current Balance
                Sheet.  This must be done after the monthly postings.

Tables used:    BAL_SHET
Forms used:     NONE
Report used:    BALSHET

************************************************************************
```

Module: PRT_INC.CMD

Menu Option: PRINT INCOME STATEMENT

Description: This module prepares and prints the Income Statement utilizing the last monthly update to the EARNINGS table. This does not account for entries made after monthly updates...only accurate as of the last posting and is normally printed at the end of the year.

The Income Statement should be printed prior to the entry of the new year transactions and reinitialization of the database.

Tables used: EARNINGS
Forms used: NONE
Report used: INCSTMT

****

5. CLUB_END.APP

****

Module: REOCCUR.CMD

Menu Option: POSTING REOCCURRING CHARGES

Description: This module post reoccurring charges to the member's charge account. The manager will note whether both the monthly dues and the member meeting charges are to be posted or just one to be posted for each member.

Tables used: MEM_REC, MEM_CHG
Forms used: NONE

****

Module: END_MON.CMD

Menu Option: POST MONTHLY JOURNAL ENTRIES

Description: This module post the monthly journal entries of most database tables and places account totals in the general purpose tables. Then the journal entries are placed in their associated historical table and the tables are prepared for the next month's transactions.

Some tables are posted and then placed in historical tables during execution of another module (eg. member charges and payments tables posted during printing of Member statements.)

Therefore, the Lessor's Statements, the Aircraft Inventory and Status Report, the Member Statements, and Instructor Statements must be posted prior to this module's operation or the posting of tables will be inaccurate.

After the postings are completed the user will be directed to perform a database back-up and repack inorder for the database to perform more efficiently.

Tables used: INCOME, EXPENSE, EARNINGS, INC_HIST, EXP_HIST, BAL_SHET, BAL_JOUR, ACCT_PAY, AP_CHG, AP_PAID, APCGHIST, APPDHIST, INS_REC, MEM_FLT, FLT_REC, FLT_HIST, A/CMAINT, MAINHIST, ACCT_REC, CAP_ASET, MBR_SUM
Forms used: NONE

****

Module: BACKPACK.CMD

Menu Option: BACK UP AND PACK THE DATABASE

Description: This module creates a backup copy of the database on floppy disks and then repacks the hard disk to reclaim any unusable space developed from removes and deletes.

Tables used: NONE
Forms used: NONE

****

Module: MEMSTMT.CMD

Menu Option: POST AND PRINT MEMBER STATEMENTS

Description: This module computes the active members' statement charges, prints the Members' Statements, updates the Club's Account Receivable records, and posts all tables preparing for the new monthly transactions.

Tables used: MEM_REC, MEM_CHG, MEM_PAY, CHGHIST, PAYHIST, BILLTEMP, ACCT_REC
Forms used: NONE
Report used: MEM_BILL

****

Module: INSTSTMT.CMD

Menu Option: POST AND PRINT INSTRUCTOR STATEMENTS

Description: This module computes and prints the Instructor Statements and updates the instructor tables for the next month.

Tables used: INST_REC, INSTHIST
Forms used: NONE
Report used: INSTSTMT

****

Module: LES_STMT.CMD

Menu Option: POST AND PRINT LESSOR STATEMENTS

Description: This module computes and prints the Lessor's Statements. The A/C_HRS and A/CMAINT tables must have been kept up to date through the month. The manager should prepare the A/C Status report after printing the Lessor's statements to ensure proper update of A/C HOURS.

Tables used: LES_ACCT, A/C_REC, A/C_HRS, A/CMAINT
Forms used: NONE

****

APPENDIX E
FLYSTART AND CLUB_MIS SOURCE CODE

The Management Information System Prototype for the Monterey Navy Flying Club consist of five applications (programs) written in R:BASE 5000 language. Each application has access to the one common R:BASE 5000 relational Database structure.

The following pages contain the source code for the first two application programs of the Monterey Navy Flying Club MIS prototype. The remaining applications are listed in Appendix F through Appendix H.

The menu selection portions of each application, which are listed first in each application, were computer generated by the R:BASE 5000 Application Express. The remaining command files (modules/routines), which comprise the applications, were written by this writer and CDR. D. R. George. Portions of the membership entry/edit transaction code, member, instructor, and lessor statement code, and database backup and repack code are from the original prototype [Ref. 1] and were incorporated into the version 2.0 prototype code.

Each command file's code is commented. Code for the Reports and Entry/Edit forms is contained within the R:BASE 5000 database structure itself and was considered inappropriate for listing.

```
*(*********************************************************************)
*(*                    MONTEREY NAVY FLYING CLUB                      *)
*(*                 MANAGEMENT INFORMATION SYSTEM                     *)
*(*                     PROTOTYPE VERSION 2.0                         *)
*(*                                                                   *)
*(*              MAIN MENU AND STARTUP APPLICATION                    *)
*(*                                                                   *)
*(*                  LCDR. JAMES M. GRAHAM, USN                       *)
*(*                         JANUARY 1987                         (07) *)
*(*********************************************************************)

$COMMAND
FLYSTART
SET MESSAGE OFF
OPEN FLYCLUB
SET ERROR MESSAGE OFF
SET VAR PICK1  INT
LABEL STARTAPP
  NEWPAGE
  CHOOSE PICK1  FROM FLYMAIN  IN FLYSTART.APX
  IF PICK1  EQ 0 THEN
    GOTO ENDAPP
  ENDIF
  IF PICK1  EQ            1 THEN
    RUN clubmis  IN FLYSTART.APX
    GOTO STARTAPP
  ENDIF
  IF PICK1  EQ            2 THEN
    RUN CLUBFIN  IN FLYSTART.APX
    GOTO STARTAPP
  ENDIF
  IF PICK1  EQ            3 THEN
    RUN FLYMAINT IN FLYSTART.APX
    GOTO STARTAPP
  ENDIF
  IF PICK1  EQ            4 THEN
    RUN AC_UPDAT IN FLYSTART.APX
    GOTO STARTAPP
  ENDIF
  IF PICK1  EQ            5 THEN
    RUN CLUBPRT  IN FLYSTART.APX
    GOTO STARTAPP
  ENDIF
  IF PICK1  EQ            6 THEN
    RUN CLUBEOMT IN FLYSTART.APX
    GOTO STARTAPP
  ENDIF
  IF PICK1  EQ            7 THEN
    GOTO ENDAPP
  ENDIF
  GOTO STARTAPP
LABEL ENDAPP
CLEAR PICK1
RETURN
$MENU
FLYMAIN
COLUMN MONTEREY NAVY FLYING CLUB MIS MAIN MENU
MEMBERSHIP TRANSACTIONS
FINANCIAL TRANSACTIONS
AIRCRAFT MAINTENANCE/FLIGHT HOUR UPDATES
AIRCRAFT INVENTORY UPDATE
PRINT REPORTS/STATEMENTS
END-OF-THE-MONTH TRANSACTIONS
RETURN TO THE COMPUTER OPERATING SYSTEM
$COMMAND
clubmis
*(*********************************************************************
PROGRAM:      CLUBMIS.CMD
AUTHOR:       J. M. GRAHAM
DATE:          NOV 1986
DESCRIPTION:   THIS ROUTINE CALLS THE FIRST APPLICATION PROGRAM FOR
               USE WITH THE FLYCLUB DATABASE OF THE MNFC MIS SYSTEM.
               THE USER WILL BE ALLOWED TO MAKE MEMBERSHIP ENTRIES/
               DELETIONS/MODIFICATIONS.

TABLES USED:   NONE
FORMS USED:    NONE
*************************************************************************)

SET MESSAGES OFF
SET ERROR MESSAGES OFF
OPEN FLYCLUB

*(* CALL THE CLUB_MIS APPLICATION *)
RUN CLUB_MIS IN CLUB_MIS.APX

*(* CALL BACK TO THE MNFC MAIN MENU *)
RETURN
*(* END OF CLUBMIS.CMD *)


$COMMAND
CLUBFIN
*(*************************************************************************
PROGRAM:       CLUBFIN.CMD
AUTHOR:        J. M. GRAHAM
DATE:          NOV 1986
DESCRIPTION:   THIS ROUTINE CALLS THE FINANCIAL APPLICATION OF THE
               MNFC MIS SYSTEM USING THE FLYCLUB DATABASE.  ALL
               FINANCIAL TRANSACTIONS ARE ENTERED IN THIS PROGRAM.

TABLES USED:   NONE
FORMS USED:    NONE
*************************************************************************)

SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

*(* CALL FINANCIAL APPLICATION *)
RUN CLUB_FIN IN CLUB_FIN.APX

*(* CALL BACK TO THE MNFC MAIN MENU *)
RETURN
*(* END OF CLUBFIN.CMD *)


$COMMAND
FLYMAINT
*(*************************************************************************
PROGRAM:       FLYMAINT.CMD
AUTHOR:        J. M. GRAHAM
DATE:          JAN 1987   VER. 1.4
DESCRIPTION:   THIS ROUTINE ALLOWS THE USER TO ENTER AIRCRAFT
               MAINTENANCE TRANSACTIONS OR UPDATE A/C FLT HRS.

TABLES USED:   A/C_HRS, A/CMAINT
FORMS USED:    MAINTFRM, FLTHRSFM                                (07)
*************************************************************************)

SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

*(* ask if maintenance or flight hour update )
LABEL ACMENU
NEWPAGE
WRITE "AIRCRAFT MAINTENANCE OR AIRCRAFT HOURS UPDATE"  AT 4,15
WRITE "-----------------------------------------------" AT 5,15
WRITE "ENTER: "  AT 7,5
WRITE "1 - POST AN AIRCRAFT MAINTENANCE TRANSACTION"  AT 9,5
WRITE "2 - POST A CHANGE/CORRECTION TO AN AIRCRAFT HOURS"  AT 11,5
WRITE "3 - RETURN TO THE MNFC MAIN MENU"  AT 13,5
WRITE "CHOICE:"  AT 15,5
FILLIN YOURCH USING " "  AT 15,15
IF YOURCH = 1 THEN
   GOTO ACMAINT
ENDIF
IF YOURCH = 2 THEN
   *(-- allows editing an existing entry in the A/C_HRS table)
FILLIN VACNUM USING "HOURS TO BE ADJUSTED ON AIRCRAFT NUMBER - "AT 20,20
   EDIT USING FLTHRSFM WHERE A/C_NUM EQ .VACNUM
   GOTO ACMENU
ENDIF
IF YOURCH = 3 THEN
   GOTO ACEND
ENDIF

LABEL ACMAINT
  LABEL MAINTMENU
  NEWPAGE
  WRITE "AIRCRAFT MAINTENANCE TRANSACTION"  AT 5,25
  WRITE "--------------------------------"  AT 6,25
  WRITE "ENTER:"  AT 8,10
  WRITE "1 - POST A NEW MAINTENANCE TRANSACTION"  AT 10,10
  WRITE "2 - POST A CHANGE TO AN EXISTING MAINTENANCE ENTRY"  AT 12,10
  WRITE "3 - EXIT TO AIRCRAFT UPDATE MENU"  AT 14,10
  WRITE "4 - QUIT AND RETURN TO THE MNFC MAIN MENU"  AT 16,10
  WRITE "CHOICE:"  AT 18,10
  FILLIN YOURANS USING " "  AT 18,20
  IF YOURANS = 1 THEN
     ENTER MAINTFRM *(-- allows for entry of maintenance transactions)
     GOTO MAINTMENU
  ENDIF
  IF YOURANS = 2 THEN
   EDIT USING MAINTFRM *(-- allows editing of maintenance transactions)
   GOTO MAINTMENU
  ENDIF
  IF YOURANS = 3 THEN
     GOTO ACMENU
  ENDIF
  IF YOURANS = 4 THEN
     GOTO ACEND
  ENDIF

   *(*  END OF ACMAINT SUBROUTINE)

LABEL ACEND
SET MESSAGES ON
SET ERROR MESSAGES ON
RETURN
*(* END OF FLYMAINT.CMD )


$COMMAND
AC_UPDAT
*(*********************************************************************
PROGRAM:      AC_UPDAT.CMD
AUTHOR:       J. M. GRAHAM
DATE:         DEC 1986   VER. 2.1
DESCRIPTION:  THIS MODULE IS USED BY THE CLUB MANAGER TO EITHER ADD
              A LEASED OR PURCHASED AIRCRAFT TO THE CLUB'S INVENTORY
              OR DELETE AN AIRCRAFT FROM THE CURRENT INVENTORY.

TABLES USED:  A/C_HRS, CAP_ASET, LES_ACT, A/C_REC
FORMS USED:   A/C_FORM, HOB_FORM, LES_FORM                      (31)
**********************************************************************)
```

```
SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

*(-- place menu choices for manager)
LABEL A/C_MENU
NEWPAGE
WRITE "AIRCRAFT INVENTORY UPDATE" AT 2,25
WRITE "_________________________" AT 3,25
WRITE "ENTER:" AT 5,10
WRITE "1 - To enter a purchased aircraft to the inventory" AT 6,10
WRITE "2 - To enter a leased aircraft to the inventory" AT 8,10
WRITE "3 - To delete a leased aircraft from inventory" AT 10,10
WRITE "4 - To delete a club owned aircraft from inventory" AT 12,10
WRITE "5 - To exit without an update" AT 14,10
WRITE "CHOICE:" AT 16,10
FILLIN YOURNO USING " " AT 16,20

*(* DETERMINE USER CHOICE AND ACTION TO TAKE *)
IF YOURNO = 1 THEN
   GOTO PURCHASE
ENDIF
IF YOURNO = 2 THEN
   GOTO LEASE
ENDIF
IF YOURNO = 3 THEN
   GOTO STOPLEAS
ENDIF
IF YOURNO = 4 THEN
   GOTO DELA/C
ENDIF
IF YOURNO = 5 THEN
   GOTO ACEND
ELSE
   WRITE "YOUR ENTRY IS INVALID" AT 12,10
   WRITE "PLEASE, TRY AGAIN...PRESS ANY KEY TO CONTINUE" AT 14,10
   PAUSE
   GOTO A/C_MENU
ENDIF

LABEL LEASE
   NEWPAGE  *(* input for new lessor data on newly acquired plane)
   ENTER LES_FORM

LABEL PURCHASE
   NEWPAGE      *(* input new a/c data on purchase )
   CLEAR VA/C_NUM
   CLEAR VTYPE
   CLEAR VYEAR
   CLEAR VMODEL
   CLEAR VPASSEAT
   CLEAR VNO_ENG
   CLEAR VCOST
   CLEAR VTX:DATE
   CLEAR VBK_VAL
   CLEAR VHULLVAL
   CLEAR VLESRATE
   CLEAR VRENT
   CLEAR VHULPREM
   CLEAR VLIABPRM
   CLEAR VPREPINS
   CLEAR VTOTPRM
   CLEAR VSTRTDAT
   CLEAR VSTOPDAT
   CLEAR VACC_DEP
   CLEAR VDEPTERM

   DRAW A/C_FORM
   WRITE "PRESS PGDN KEY TO ENTER DATA - PRESS ESC KEY TO QUIT" AT 24,8
   ENTER VARIABLE VA/C_NUM VTYPE VYEAR VMODEL VPASSEAT VNO_ENG VCOST +
         VTX:DATE VBK_VAL VHULLVAL VLESRATE VRENT VHULPREM VLIABPRM +
         VPREPINS VTOTPRM VSTRTDAT VACC_DEP VSTOPDAT VDEPTERM +
         RETURN PGDN ESC
   IF #RETURN = ESC THEN
      BREAK
   ENDIF

   LOAD A/C_REC
      .VA/C_NUM .VTYPE .VYEAR .VMODEL .VPASSEAT .VNO_ENG .VHULLVAL +
      .VHULPREM .VLIABPRM .VTOTPRM .VSTRTDAT .VSTOPDAT .VLESRATE +
      .VRENT .VPREPINS .VBK_VAL .VACC_DEP .VDEPTERM .VTX:DATE
   END

      *(* prepare var to enter new aircraft in capital asset acct)
   SET VARIABLE VASETNO TO "1650"
   SET VARIABLE VWHAT1 TO "AIRPLANE"
   SET VARIABLE VWHAT2 TO .VA/C_NUM & .VTYPE
   SET VARIABLE VWHAT TO .VWHAT1 & .VWHAT2

   LOAD CAP_ASET
       .VWHAT .VASETNO .VCOST .VTX:DATE .VDEPTERM .VACC_DEP
   END

   NEWPAGE       *(* input a/c hours for new aircraft )
   CLEAR VA/C_NUM
   CLEAR VHOBSTAR
   CLEAR VHOB_END
   CLEAR VCUMHRS
   CLEAR VHOBMAIN
   CLEAR VHOBOWNR

   DRAW HOB_FORM
   WRITE "PRESS PGDN KEY TO ENTER DATA - PRESS ESC KEY TO QUIT" AT 22,8
   ENTER VARIABLE VA/C_NUM VHOBSTAR VHOB_END VCUMHRS VHOBMAIN VHOBOWNR +
         RETURN PGDN ESC
   IF #RETURN = ESC THEN
      BREAK
   ENDIF

   SET VARIABLE VHOBUSE TO 0.0
   SET VARIABLE VHOBNET TO 0.0
   SET VARIABLE VHBLEAS TO 0.0
   SET VARIABLE VTOTMAIN TO $0.00

   LOAD A/C_HRS
        .VA/C_NUM .VCUMHRS .VHOB_END .VHOBSTAR .VHOBUSE .VHOBMAIN +
        .VHOBNET .VHOBOWNR .VHBLEAS .VLESRATE .VTOTMAIN
   END
   GOTO ACEND

LABEL STOPLEAS
   NEWPAGE
   WRITE "WARNING !!!!" AT 5,25
   WRITE "All aircraft deletions should ONLY be performed after" AT 8,5
   WRITE "the end-of-month transactions and all billing are" AT 10,5
   WRITE "accomplished.  Otherwise, all aircraft flight hours" AT 12,5
   WRITE "and maintenance cost for the month are lost on that" AT 14,5
   WRITE "deleted aircraft."  AT 16,5
   WRITE "DO YOU DESIRE TO STILL DELETE THE AIRCRAFT NOW(Y/N)?" AT 20,5
   FILLIN YOURCH USING " "  AT 20,60
   IF YOURCH = N THEN
      GOTO ACEND
   ENDIF

   NEWPAGE  *(* input lessor's no. or name to delete from database)
   WRITE "PLEASE, EITHER THE LESSOR'S NUMBER OR FULL NAME" AT 5,10
   WRITE "ONLY INPUT ONE !"  AT 7,30
   FILLIN VLESOR USING "LESSOR'S NUMBER IS - "  AT 10,20
   FILLIN VLNAME USING "LESSOR'S LAST NAME IS - " AT 12,20
   IF VLESOR EXISTS THEN
      NEWPAGE
      EDIT USING LES_FORM WHERE LESOR_NU = .VLESOR
   ELSE
      NEWPAGE
      EDIT USING LES_FORM WHERE L:NAME = .VLNAME
   ENDIF

   NEWPAGE *(* input a/c no. inorder to delete all rows concerning it)
   WRITE "A/C Number, Please:" AT 2,20
   FILLIN VA/C_NUM USING " " AT 2,40
   WRITE "DELETION OF AIRCRAFT IS NOW TAKING PLACE"  AT 10,15

   *(* remove all records of this aircraft from the a/c_rec table)
   DELETE ROWS FROM A/C_REC WHERE A/C_NUM = .VA/C_NUM
   *(* remove all records of this aircraft from the a/c_hrs table)
   DELETE ROWS FROM A/C_HRS WHERE A/C_NUM = .VA/C_NUM
   *(* remove all records of this aircraft from the maint table)
   DELETE ROWS FROM A/CMAINT WHERE A/C_NUM = .VA/C_NUM
   WRITE "DELETION COMPLETED - PRESS ANY KEY TO CONTINUE" AT 15,15
   PAUSE
   GOTO ACEND  *(* end of stoplease routine)

LABEL DELA/C
 NEWPAGE
 WRITE "WARNING !!!!"  AT 5,25
 WRITE "All aircraft deletions should ONLY be performed after"  AT 8,5
 WRITE "the end-of-month transactions and all billing are" AT 10,5
 WRITE "accomplished.  Otherwise, all aircraft flight hours and"AT 12,5
 WRITE "maintenance cost for the month are lost on that"  AT 14,5
 WRITE "deleted aircraft."  AT 16,5
 WRITE "DO YOU DESIRE TO STILL DELETE THE AIRCRAFT NOW(Y/N)?"  AT 20,5
 FILLIN YOURCH USING " "  AT 20,60
 IF YOURCH = N THEN
    GOTO ACEND
 ENDIF

 NEWPAGE  *(* input a/c no. and type to remove from database)
 WRITE "A/C Number and A/C Type, Please:"  AT 2,20
 FILLIN VA/C_NUM USING "A/C Number is - " AT 4,10
 FILLIN VTYPE USING "A/C Type is - "  AT 6,10
WRITE "DELETION OF AIRCRAFT FROM DATABASE IS NOW TAKING PLACE" AT 10,15

   *(* prepare search data to delete the proper capital asset )
   SET VARIABLE VWHAT1 TO "AIRPLANE"
   SET VARIABLE VWHAT2 TO .VA/C_NUM & .VTYPE
   SET VARIABLE VIDENT TO .VWHAT1 & .VWHAT2

   *(* remove all records of this aircraft from cap_aset table)
   DELETE ROWS FROM CAP_ASET WHERE IDENT = .VIDENT
   *(* remove all records of this aircraft from a/c_rec table)
   DELETE ROWS FROM A/C_REC WHERE A/C_NUM = .VA/C_NUM
   *(* remove all records of this aircraft from a/c_hrs table)
   DELETE ROWS FROM A/C_HRS WHERE A/C_NUM = .VA/C_NUM
   *(* remove all records of this aircraft from a/cmaint table)
   DELETE ROWS FROM A/CMAINT WHERE A/C_NUM = .VA/C_NUM
   WRITE "DELETION COMPLETED - PRESS ANY KEY TO CONTINUE"  AT 15,15
   PAUSE

LABEL ACEND
SET MESSAGES ON
SET ERROR MESSAGES ON
RETURN
*(* END OF AC_UPDAT.CMD *)
$COMMAND
CLUBPRT
*(**********************************************************************
PROGRAM:       CLUBPRT.CMD
AUTHOR:        J. M. GRAHAM
DATE:          NOV 1986
DESCRIPTION:   THIS ROUTINE CALLS THE CLUB PRINTING APPLICATION
               FOR THE MNFC MIS SYSTEM USING THE FLYCLUB DATABASE.

TABLES USED:   NONE
FORMS USED:    NONE
**********************************************************************)

SET MESSAGES OFF
SET ERROR MESSAGES OFF
OPEN FLYCLUB

*(* CALL CLUBPRINT *)
RUN CLUB_PRT IN CLUB_PRT.APX

*(* CALL BACK TO MNFC MAIN MENU *)
RETURN
*(* END OF CLUBPRT.CMD *)

$COMMAND
CLUBEOMT
*(*********************************************************************
PROGRAM:       CLUBEOMT.CMD
AUTHOR:        J. M. GRAHAM
DATE:          NOV 1986
DESCRIPTION:   THIS ROUTINE CALLS THE END-OF-THE-MONTH APPLICATION
               FOR THE MNFC MIS SYSTEM USING THE FLYCLUB DATABASE.
               ALL ACCOUNTS ARE POSTED AND UPDATED MONTHLY.  ALL
               REOCCURRING FINANCIAL TRANSACTIONS ARE CONDUCTED AT
               THIS TIME.

TABLES USED:   NONE
FORMS USED:    NONE
**********************************************************************)

SET MESSAGES OFF
SET ERROR MESSAGES OFF
OPEN FLYCLUB

*(* CALL END-OF-MONTH TRANSACTIONS APPLICATION *)
RUN CLUB_END IN CLUB_END.APX

*(* CALL BACK TO THE MNFC MAIN MENU *)
RETURN
*(* END OF CLUBEOMT.CMD *)

$COMMAND
CLUB_MIS
*(*********************************************************************)
*(*                  MONTEREY NAVY FLYING CLUB                       *)
*(*                MANAGEMENT INFORMATION SYSTEM                     *)
*(*                   PROTOTYPE VERSION 2.0                          *)
*(*                                                                  *)
*(*               MEMBERSHIP TRANSACTIONS PROGRAM                    *)
*(*                                                                  *)
*(*                  LCDR. JAMES M. GRAHAM, USN                      *)
*(*                        JANUARY 1987                         (07) *)
*(*********************************************************************)

SET MESSAGE OFF
OPEN FLYCLUB
SET ERROR MESSAGE OFF
*(-- menu selection routine; computer generated code )

SET VAR PICK1  INT
LABEL STARTAPP
  NEWPAGE
  CHOOSE PICK1  FROM MIS_MAIN IN CLUB_MIS.APX
  IF PICK1  EQ 0 THEN
    GOTO ENDAPP
  ENDIF
  IF PICK1  EQ            1 THEN
    RUN ADD_MEM  IN CLUB_MIS.APX
    GOTO STARTAPP
  ENDIF
  IF PICK1  EQ            2 THEN
    RUN EDIT_MEM IN CLUB_MIS.APX
    GOTO STARTAPP
  ENDIF
  IF PICK1  EQ            3 THEN
    GOTO ENDAPP
  ENDIF
  GOTO STARTAPP
LABEL ENDAPP
CLEAR PICK1
RETURN

*(-- menu screen routine )
$MENU
MIS_MAIN
COLUMN MEMBERSHIP TRANSACTIONS
ENTER A NEW MEMBER RECORD
EDIT / DELETE AN EXISTING MEMBER RECORD
RETURN TO THE MNFC MAIN MENU

$COMMAND
ADD_MEM
*(*******************************************************************
PROGRAM:      ADD_MEM.CMD   VER.2.0
AUTHOR:       J. M. GRAHAM
DATE:         NOV 1986
DESCRIPTION:  THIS MODULE ALLOWS THE USER TO ENTER A NEW MEMBER'S
              DATA UTILIZING A PREDESIGNED SCREEN INPUT FORMAT.

TABLES USED:  MEM_REC
FORMS USED:   MEM_FORM
*******************************************************************)
SET MESSAGES OFF
SET ERROR MESSAGES OFF
OPEN FLYCLUB
*(* ACCESS TO THE INPUT FORM AND ACCEPTING MEMBER DATA *)
ENTER MEM_FORM
*(* RESET ALL ERROR MESSAGE CAPABILITY *)
SET MESSAGES ON
SET ERROR MESSAGES ON
*(* ALLOW RETURN TO MENU )
RETURN
*(* END OF ADD_MEM.CMD *)
$COMMAND
EDIT_MEM
*(*******************************************************************
PROGRAM:      EDIT_MEM.CMD
AUTHOR:       J. M. GRAHAM
DATE:         JAN 1987
DESCRIPTION:  THIS MODULE ALLOWS THE USER TO EDIT OR DELETE MEMBER
              DATA AND/OR AN ENTIRE MEMBER RECORD.  THE USER ACCESS
              THE MEM_REC TABLE USING THE MEM_FORM FORM.  SPECIFIC
              RECORD IS CALLED EITHER BY MBR # OR MBR LAST NAME.

TABLES USED:  MEM_REC
FORMS USED:   MEM_FORM                                         (07)
*******************************************************************)

SET MESSAGES OFF
SET ERROR MESSAGES OFF
OPEN FLYCLUB
```

```
*(* PREPARE SCREEN FOR ASKING FOR INPUT DATA REQUIRED TO ACCESS FORM)
NEWPAGE
WRITE "ENTER, ONCE PROMPTED, EITHER THE MEMBER'S NUMBER," AT 5,10
WRITE "OR, THE MEMBER'S LAST NAME OF WHOM YOU WISH TO EDIT" AT 6,10
WRITE " ***** ENTER ONLY ONE...NOT BOTH MEMBER NUMBER AND NAME *****"
     AT 9,5
LABEL GETDATA
  FILLIN MEMNUM USING "MEMBER NUMBER:" AT 15,10
  FILLIN MNAME USING "MEMBER'S LAST NAME:" AT 17,10

  IF MEMNUM EXISTS AND MNAME EXISTS THEN
    WRITE "YOU CAN ONLY ENTER EITHER THE MEMBER'S NUMBER OR NAME"
       AT 20,20
    WRITE "PRESS ANY KEY TO REENTER YOUR DATA"  AT 22,30
    PAUSE
    NEWPAGE
    GOTO GETDATA
  ENDIF

*(* DETERMINE USER CHOICE AND ACT AS REQUESTED *)
IF MEMNUM FAILS THEN
   EDIT USING MEM_FORM SORTED BY L:NAME MEM_NUM WHERE L:NAME EQ .MNAME
   CLEAR MNAME
   CLEAR MEMNUM
ELSE
 EDIT USING MEM_FORM SORTED BY MEM_NUM L:NAME WHERE MEM_NUM EQ .MEMNUM
 CLEAR MNAME
 CLEAR MEMNUM
ENDIF
*(* CALL MEMU *)
RETURN
*(* END OF EDIT_MEM.CMD *)
```

APPENDIX F
CLUB_FIN SOURCE CODE

```
*(***********************************************************************)
*(*                   MONTEREY NAVY FLYING CLUB                         *)
*(*                 MANAGEMENT INFORMATION SYSTEM                       *)
*(*                    PROTOTYPE VERSION 2.0                            *)
*(*                                                                     *)
*(*               FINANCIAL TRANSACTIONS APPLICATION                    *)
*(*                                                                     *)
*(*                   LCDR. JAMES M. GRAHAM, USN                        *)
*(*                        JANUARY 1987                            (07) *)
*(***********************************************************************)

$COMMAND
CLUB_FIN
SET MESSAGE OFF
OPEN FLYCLUB
SET ERROR MESSAGE OFF
SET VAR PICK1  INT
LABEL STARTAPP
  NEWPAGE
  CHOOSE PICK1  FROM FIN_MAIN IN CLUB_FIN.APX
  IF PICK1  EQ 0 THEN
    GOTO ENDAPP
  ENDIF
  IF PICK1  EQ            1 THEN
    SET VAR PICK2  INT
    SET VAR LEVEL2 INT
    SET VAR LEVEL2 TO 0
    WHILE LEVEL2 EQ 0  THEN
      NEWPAGE
      CHOOSE PICK2  FROM MGR_MENU IN CLUB_FIN.APX
      IF PICK2  EQ 0 THEN
        BREAK
      ENDIF
      IF PICK2  EQ            1 THEN
        SET VAR PICK3  INT
        SET VAR LEVEL3 INT
        SET VAR LEVEL3 TO 0
        WHILE LEVEL3 EQ 0  THEN
          NEWPAGE
          CHOOSE PICK3  FROM BAL_SHET IN CLUB_FIN.APX
          IF PICK3  EQ 0 THEN
            BREAK
          ENDIF
          IF PICK3  EQ            1 THEN
            RUN ASSTEDIT IN CLUB_FIN.APX
          ENDIF
          IF PICK3  EQ            2 THEN
            RUN LIABEDIT IN CLUB_FIN.APX
          ENDIF
          IF PICK3  EQ            3 THEN
            BREAK
          ENDIF
        ENDWHILE
        CLEAR LEVEL3
        CLEAR PICK3
      ENDIF
      IF PICK2  EQ            2 THEN
        SET VAR PICK3  INT
        SET VAR LEVEL3 INT
        SET VAR LEVEL3 TO 0
```

```
          WHILE LEVEL3 EQ 0  THEN
            NEWPAGE
            CHOOSE PICK3  FROM INCSTMT  IN CLUB_FIN.APX
            IF PICK3  EQ 0 THEN
              BREAK
            ENDIF
            IF PICK3  EQ            1 THEN
              RUN REV_EDIT IN CLUB_FIN.APX
            ENDIF
            IF PICK3  EQ            2 THEN
              RUN EXP_EDIT IN CLUB_FIN.APX
            ENDIF
            IF PICK3  EQ            3 THEN
              BREAK
            ENDIF
          ENDWHILE
          CLEAR LEVEL3
          CLEAR PICK3
        ENDIF
        IF PICK2  EQ            3 THEN
          BREAK
        ENDIF
      ENDWHILE
      CLEAR LEVEL2
      CLEAR PICK2
      GOTO STARTAPP
    ENDIF
    IF PICK1  EQ            2 THEN
      SET VAR PICK2  INT
      SET VAR LEVEL2 INT
      SET VAR LEVEL2 TO 0
      WHILE LEVEL2 EQ 0  THEN
        NEWPAGE
        CHOOSE PICK2  FROM NEW_FIN  IN CLUB_FIN.APX
        IF PICK2  EQ 0 THEN
          BREAK
        ENDIF
        IF PICK2  EQ            1 THEN
          RUN PUTCHG   IN CLUB_FIN.APX
        ENDIF
        IF PICK2  EQ            2 THEN
          RUN PUTPAY   IN CLUB_FIN.APX
        ENDIF
        IF PICK2  EQ            3 THEN
          RUN CRED_BUY IN CLUB_FIN.APX
        ENDIF
        IF PICK2  EQ            4 THEN
          RUN CASH_BUY IN CLUB_FIN.APX
        ENDIF
        IF PICK2  EQ            5 THEN
          RUN PD_ACCT  IN CLUB_FIN.APX
        ENDIF
        IF PICK2  EQ            6 THEN
          BREAK
        ENDIF
      ENDWHILE
      CLEAR LEVEL2
      CLEAR PICK2
      GOTO STARTAPP
    ENDIF
    IF PICK1  EQ            3 THEN
      SET VAR PICK2  INT
      SET VAR LEVEL2 INT
      SET VAR LEVEL2 TO 0
      WHILE LEVEL2 EQ 0  THEN
        NEWPAGE
        CHOOSE PICK2  FROM EDIT_FIN IN CLUB_FIN.APX
        IF PICK2  EQ 0 THEN
          BREAK
      ENDIF
      IF PICK2  EQ             1 THEN
        RUN EDIT_CHG IN CLUB_FIN.APX
      ENDIF
      IF PICK2  EQ             2 THEN
        RUN EDIT_PAY IN CLUB_FIN.APX
      ENDIF
      IF PICK2  EQ             3 THEN
        RUN EDIT_AP  IN CLUB_FIN.APX
      ENDIF
      IF PICK2  EQ             4 THEN
        RUN EDIT_PD  IN CLUB_FIN.APX
      ENDIF
      IF PICK2  EQ             5 THEN
        BREAK
      ENDIF
    ENDWHILE
    CLEAR LEVEL2
    CLEAR PICK2
    GOTO STARTAPP
  ENDIF
  IF PICK1  EQ             4 THEN
    GOTO ENDAPP
  ENDIF
  GOTO STARTAPP
LABEL ENDAPP
CLEAR PICK1
RETURN
$MENU
FIN_MAIN
COLUMN FINANCIAL TRANSACTIONS
MANAGER'S UPDATE ENTRY
ENTER A NEW TRANSACTION
EDIT / DELETE AN EXISTING TRANSACTION
RETURN TO THE MNFC MAIN MENU
$MENU
MGR_MENU
COLUMN MANAGER'S UPDATE ENTRY
BALANCE SHEET ADJUSTMENT
INCOME STATEMENT ADJUSTMENT
RETURN TO FINANCIAL TRANSACTION MENU
$MENU
NEW_FIN
COLUMN ENTER A NEW FINANCIAL TRANSACTION
ENTER A MEMBER'S CHARGES
ENTER A MEMBER'S PAYMENTS
POST AN ACCOUNTS PAYABLE PURCHASE/INTEREST
POST CASH PURCHASE OF SERVICES/GOODS
POST PAYMENT TO AN ACCOUNTS PAYABLE ACCOUNT
RETURN TO FINANCIAL TRANSACTIONS MENU
$MENU
EDIT_FIN
COLUMN EDIT / DELETE AN EXISTING TRANSACTION
MEMBER CHARGE TRANSACTION
MEMBER PAYMENT TRANSACTION
ACCOUNTS PAYABLE PURCHASE
PAYMENT TO AN ACCOUNTS PAYABLE ACCOUNT
RETURN TO FINANCIAL TRANSACTIONS MENU
$MENU
INCSTMT
COLUMN INCOME STATEMENT ADJUSTMENT
REVENUE / SALE ACCOUNT ENTRY
EXPENSE ACCOUNT ENTRY
RETURN TO MANAGER'S UPDATE ENTRY MENU
$MENU
BAL_SHET
COLUMN BALANCE SHEET ADJUSTMENT
ASSET ACCOUNT ENTRY
LIABILITY ACCOUNT ENTRY
```

```
RETURN TO MANAGER'S UPDATE ENTRY MENU
$COMMAND
PUTCHG
*(**********************************************************************
PROGRAM:      PUTCHG.CMD
AUTHOR:       J. M. GRAHAM
DATE:         JAN  1986   VER 1.6
DESCRIPTION:  ENTERS MEMBER'S CHARGES (INPUT BY MANAGER
              FROM INDIVIDUAL CHARGE SHEETS).  AND THIS MODULE UPDATES
              THE TOTAL AIRCRAFT HOURS USED AND CUM_HRS OF THE A/C.

TABLES USED:  MEM_REC, MEM_CHG, MEM_FLT, INST_REC, INCOME, EXPENSE,
              A/C_REC, A/C_HRS, FLT_REC
FORM USED:    CHGFORM                                            (15)
***********************************************************************

*(- Establish message parameters)
SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

*(Establish loop for entering member number)
SET VARIABLE VMEMNUM TEXT
SET VARIABLE VMEMNUM TO 0
WHILE VMEMNUM EXISTS THEN
   NEWPAGE
   CLEAR VINVONUM
   CLEAR VTRANDAT
   CLEAR VACCHARG
   CLEAR VHRSFLON
   CLEAR VACNUM
   CLEAR VINSTNAM
   CLEAR VINCHARG
   CLEAR VFSCHARG
   CLEAR VMSCHARG
   CLEAR VMSCREDT
   CLEAR VEXPCHG
   CLEAR VCRTYPE
   CLEAR VCHGTYPE
   CLEAR VFLT_HRS
   CLEAR VCATG
   CLEAR VCERT
   CLEAR VHOBSTRT
   CLEAR VHOBEND
   CLEAR VACCTNUM

   *(Get member number)
   WRITE "Add New Charges    (ESC to quit)" AT 2,1
   FILLIN VMEMNUM USING "Enter member number - " AT 4,1
   *(If no member number entered, exit)
   IF VMEMNUM FAILS THEN
      SET MESSAGES ON
      SET ERROR MESSAGES ON
      BREAK
   ENDIF *(VMEMNUM fails)
   *(Look up member number in MEM_REC table)
   SET POINTER #1 STATUS1 FOR MEM_REC WHERE MEM_NUM = .VMEMNUM
   *(If member number not found, warn user)
   IF STATUS1 <> 0 THEN
      WRITE "No such member number!" AT 18,1
      WRITE "Press any key to continue." AT 19,1
      PAUSE
      BREAK
   *(Otherwise set up variables for screen)
   ELSE
    SET VARIABLE NAME1 TO L:NAME IN #1
    SET VARIABLE NAME2 TO .NAME1 + ","
    SET VARIABLE NAME3 TO F:NAME IN #1
    SET VARIABLE NAME4 TO .NAME2 & .NAME3
```

```
     SET VARIABLE NAME5 TO MID_INIT IN #1
     SET VARIABLE VFULLNAM TO .NAME4 & .NAME5
    SET VARIABLE VCATG TO CATEGORY IN #1
    SET VARIABLE VCERT TO FAA_CERT IN #1
    SET VARIABLE VEXFUEL TO "EXCESS FUEL CHARGE"
    SET VARIABLE VINITFEE TO "INITIATION FEE"
    SET VARIABLE VKEYDEP TO "KEY DEPOSIT"
    SET VARIABLE VINITDUE TO "INITIAL DUES"
    SET VARIABLE VFUELCR TO "FUEL CREDIT"
    SET VARIABLE VKEYRET TO "KEY RETURN"
    SET VARIABLE VDUESCR TO "DUES CREDIT"
    SET VARIABLE VACRENT TO "A/C RENTAL"
    SET VARIABLE VINCHG TO "CFI: "
    SET VARIABLE VFSCHG TO "FLIGHT SUPPLIES"
   SET VARIABLE VFLT_HRS TO FLT_HRS IN MEM_FLT WHERE MEM_NUM EQ .VMEMNUM

    ENDIF  *( Status1 <> 0)
    NEWPAGE
    DRAW CHGFORM WITH VARIABLE VMEMNUM VFULLNAM VFLT_HRS
   WRITE "PRESS PGDN KEY TO ENTER DATA - PRESS ESC KEY TO QUIT" AT 24,8
    ENTER VARIABLE VACNUM VINVONUM VTRANDAT VHOBEND VHOBSTRT VHRSFLON +
      VINSTNUM VINCHARG VFSCHARG VCHGTYPE VMSCHARG VCRTYPE VMSCREDT +
      RETURN PGDN ESC
    *(If escape, break loop)
    IF #RETURN = ESC THEN
       BREAK
    ENDIF  *(Escape)
    *(Look up aircraft number)
    SET POINTER #2 STATUS2 FOR A/C_REC WHERE A/C_NUM = .VACNUM
    *( If aircraft number found, compute rental charge)
    IF STATUS2 = 0 THEN
       SET VARIABLE VRENT TO RENT_RAT IN #2
       SET VARIABLE VACCHARG TO .VHRSFLON X .VRENT
    ENDIF *(Status2 = 0)
    *(- Look up instructor name)
    SET POINTER #3 STATUS3 FOR MEM_REC WHERE INST_NUM = .VINSTNUM
    IF STATUS3 = 0 THEN
       SET VARIABLE VINSTNAM TO L:NAME IN #3
    ENDIF *(- look up instructor)
    NEWPAGE
    DRAW CHGFORM WITH ALL
    WRITE " PRESS ANY KEY TO CONTINUE - ESC TO QUIT " AT 24,15
    PAUSE
    IF VACCHARG <> 0 THEN  *(AIRCRAFT RENTAL)
      SET VARIABLE VRENTAL TO .VACRENT & .VACNUM
      LOAD MEM_CHG
          .VMEMNUM .VINVONUM .VTRANDAT .VACCHARG .VRENTAL +
          .VHRSFLON .VACNUM
      END  *(Load )
      SET VARIABLE VACCTNUM TO "4951"  *(-- load income from a/c rental)
      LOAD INCOME
          .VACCTNUM .VRENTAL .VACCHARG .VTRANDAT P
      END
*(-- Update cum flight hours, last date flown, and last aircraft flown)

      IF VHRSFLON EXISTS THEN
         LOAD FLT_REC
             .VTRANDAT .VHRSFLON .VCATG .VCERT .VACNUM
         END *( load )
         SET VARIABLE VFLT_HRS TO .VFLT_HRS + .VHRSFLON
         LOAD MEM_FLT
             .VMEMNUM .VCATG .VCERT .VFLT_HRS .VTRANDAT .VACNUM
         END *(-- Load )
     *(-- compute and enter new cum hrs of aircraft flown)
       SET VARIABLE VHRSUSED TO .VHOBEND - .VHOBSTRT
      SET VARIABLE VCUMHRS TO CUM_HRS IN A/C_HRS WHERE A/C_NUM = .VACNUM
       SET VARIABLE VTHOURS TO .VCUMHRS + .VHRSUSED
       CHANGE CUM_HRS TO .VTHOURS IN A/C_HRS WHERE A/C_NUM = .VACNUM
       CHANGE HOBUSED TO .VHRSUSED IN A/C_HRS WHERE A/C_NUM = .ACNUM
     CHANGE HOB_END TO .VHOBEND IN A/C_HRS WHERE A/C_NUM = .VACNUM
     CHANGE HOBSTART TO .VHOBSTRT IN A/C_HRS WHERE A/C_NUM = .VACNUM
   ENDIF *(- VHRSFLON exists )

 ENDIF *(Aircraft rental)

  IF VINCHARG <> 0 THEN  *(Instruction charge)
      *(-- Load INST_REC table for generating instructor statements)
      SET VARIABLE VCFICHG TO .VINCHG & .VINSTNAM
      LOAD INST_REC
          .VINSTNUM .VTRANDAT .VMEMNUM .VHRSFLON .VACNUM .VINCHARG +
          .VINVONUM
      END *(-- Load )
      LOAD MEM_CHG
          .VMEMNUM .VINVONUM .VTRANDAT .VINCHARG .VCFICHG +
          .VHRSFLON .VACNUM
      END  *(Load)
      SET VARIABLE VACCTNUM TO "4952"
      LOAD INCOME *(-- load income from instruction given)
          .VACCTNUM .VCFICHG .VINCHARG .VTRANDAT P
      END
  ENDIF  *(End instruction charge)

  SET VARIABLE VACNUM TO " "
  SET VARIABLE VHRSFLON TO " "
  IF VFSCHARG <> 0 THEN  *(Flight supplies charge)
      LOAD MEM_CHG
          .VMEMNUM .VINVONUM .VTRANDAT .VFSCHARG .VFSCHG +
          .VHRSFLON .VACNUM
      END  *(Load)
      SET VARIABLE VACCTNUM TO "4941"
      LOAD INCOME *(-- load income from sale of flight supplies)
          .VACCTNUM .VFSCHG .VFSCHARG .VTRANDAT P
      END
  ENDIF  *(End flight supplies charge)
  IF VCHGTYPE = 1 THEN  *(Excess fuel charge)
      LOAD MEM_CHG
          .VMEMNUM .VINVONUM .VTRANDAT .VMSCHARG .VEXFUEL +
          .VHRSFLON .VACNUM
      END  *(Load )
      SET VARIABLE VACCTNUM TO "8192"
      LOAD INCOME *(-- load income from fuel charges)
          .VACCTNUM .VEXFUEL .VMSCHARG .VTRANDAT P
      END
  ENDIF *(Excess fuel)
  IF VCHGTYPE = 2 THEN  *(Initiation fee)
      LOAD MEM_CHG
          .VMEMNUM .VINVONUM .VTRANDAT .VMSCHARG .VINITFEE +
          .VHRSFLON .VACNUM
      END  *(Load )
      SET VARIABLE VACCTNUM TO "8101"
      LOAD INCOME  *(-- load income from initiation fees)
          .VACCTNUM .VINITFEE .VMSCHARG .VTRANDAT P
      END
  ENDIF *(Initiation fee)
  IF VCHGTYPE = 3 THEN  *(Key deposit)
      LOAD MEM_CHG
          .VMEMNUM .VINVONUM .VTRANDAT .VMSCHARG .VKEYDEP +
          .VHRSFLON .VACNUM
      END  *(Load )
      SET VARIABLE VACCTNUM TO "8191"
      LOAD INCOME *(-- load income from key deposits )
          .VACCTNUM .VKEYDEP .VMSCHARG .VTRANDAT P
      END
  ENDIF *(Key deposit)
  IF VCHGTYPE = 4 THEN  *(Initial dues)
      LOAD MEM_CHG
          .VMEMNUM .VINVONUM .VTRANDAT .VMSCHARG .VINITDUE +
          .VHRSFLON .VACNUM
```

```
        END  *(Load )
        SET VARIABLE VACCTNUM TO "8100"
        LOAD INCOME  *(-- load income from initial dues)
            .VACCTNUM .VINITDUE .VMSCHARG .VTRANDAT P
        END
    ENDIF *(Initial dues)

    SET VARIABLE VEXPCHG TO .VMSCREDT
    SET VARIABLE VMSCREDT = .VMSCREDT X -1
    IF VCRTYPE = 5 THEN  *(Fuel credit)
        LOAD MEM_CHG
            .VMEMNUM .VINVONUM .VTRANDAT .VMSCREDT .VFUELCR  +
            .VHRSFLON .VACNUM
        END  *(Load )
        SET VARIABLE VACCTNUM TO "7703"
        LOAD EXPENSE  *(-- load expense from refunds of gas)
            .VACCTNUM .VFUELCR .VEXPCHG .VTRANDAT P
        END
    ENDIF *(Fuel credit)
    IF VCRTYPE = 6 THEN  *(Key return)
        LOAD MEM_CHG
            .VMEMNUM .VINVONUM .VTRANDAT .VMSCREDT .VKEYRET +
            .VHRSFLON .VACNUM
        END  *(Load )
        SET VARIABLE VACCTNUM TO "7711"
        LOAD EXPENSE  *(--  load expense of refund for key returns)
            .VACCTNUM .VKEYRET .VEXPCHG .VTRANDAT P
        END
    ENDIF *(Key return)
    IF VCRTYPE = 7 THEN  *(Dues credit)
        LOAD MEM_CHG
            .VMEMNUM .VINVONUM .VTRANDAT .VMSCREDT .VDUESCR +
            .VHRSFLON .VACNUM
        END  *(Load )
        SET VARIABLE VACCTNUM TO "7711"
        LOAD EXPENSE  *(--  load expense due to dues' credit)
            .VACCTNUM .VDUESCR .VEXPCHG .VTRANDAT P
        END
    ENDIF *(Dues credit)
ENDWHILE

*(* return to financial transaction main menu)
SET MESSAGES ON
SET ERROR MESSAGES ON
RETURN
*(* END OF PUTCHG.CMD )

$COMMAND
PUTPAY
*(*********************************************************************
PROGRAM:       PUTPAY.CMD
AUTHOR:        J. M. GRAHAM
DATE:          DEC 1986   VER 1.2
DESCRIPTION:   ENTERS MEMBER'S PAYMENTS (INPUT BY MANAGER)

TABLES USED:   MEM_REC, MEM_PAY, BAL_JOUR
FORM USED:     PAYFORM
*********************************************************************)

SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

*(* inform user that multiple entries can be made )
NEWPAGE
WRITE "MNFC MEMBER PAYMENT ENTRY"   AT 5,25
WRITE "-------------------------"  AT 6,25
WRITE"YOU WILL BE ABLE TO MAKE ALL YOUR MEMBER PAYMENT ENTRIES"AT10,10
WRITE "DURING THIS ONE SITTING USING THIS MEMU SELECTION"  AT 11,10
```

```
WRITE"YOU WILL HAVE TO USE THE MEMBER'S NUMBER TO ENTER THE PAYMENT"
     AT 13,10
WRITE"PRESS THE ESC KEY TO QUIT AND RETURN TO THE FINANCIAL MENU"
     AT 14,10
WRITE "HAVE YOUR PAYMENT DATA AND MEMBER NUMBERS READY"  AT 16,15
WRITE "NOW PRESS ANY KEY TO START ENTRY"  AT 18,20
PAUSE

LABEL GETNUM
SET VARIABLE VMEMNUM TO 0
WHILE VMEMNUM EXISTS THEN
   NEWPAGE
   CLEAR VCHKNUM
   CLEAR VTRANDAT
   CLEAR VTRANAMT
   *(Get member number)
   WRITE "Add New Payments      (ESC to Quit)" AT 2,20
   FILLIN VMEMNUM USING "The Member's Number is - " AT 4,10
   *(If no member number entered, exit)
   IF VMEMNUM FAILS THEN
      SET MESSAGES ON
      SET ERROR MESSAGES ON
      BREAK
   ENDIF  *(vmemnum fails)
   *(Look up member number in MEM_REC table)
   SET POINTER #1 STATUS1 FOR MEM_REC WHERE MEM_NUM = .VMEMNUM
   *( If member number not found, warn user)
   IF STATUS1 <> 0 THEN
      WRITE "No such member number!" AT 18,20
      WRITE "Press any key to continue." AT 19,20
      PAUSE
      GOTO GETNUM
   ELSE
      SET VARIABLE NAME1 TO L:NAME IN #1
      SET VARIABLE NAME2 TO .NAME1 + ","
      SET VARIABLE NAME3 TO F:NAME IN #1
      SET VARIABLE NAME4 TO .NAME2 & .NAME3
      SET VARIABLE NAME5 TO MID_INIT IN #1
      SET VARIABLE VFULLNAM TO .NAME4 & .NAME5
      SET VARIABLE VTRANDAT TO .#DATE
   ENDIF   *(Status1 <> 0)
   NEWPAGE
   DRAW PAYFORM WITH VARIABLE VMEMNUM VFULLNAM VTRANDAT
WRITE"PRESS PAGE DOWN KEY TO ENTER DATA-PRESS ESC KEY TO QUIT"AT 22,8
   ENTER VARIABLE VCHKNUM VPAYMENT RETURN PGDN ESC
   *(If escape, break loop)
   IF #RETURN = ESC THEN
   BREAK
   ENDIF  *(Escape)
   LOAD MEM_PAY
     .VMEMNUM .VCHKNUM .VTRANDAT .VPAYMENT PAYMENT
   END  *(End of Load)

   *(* show increase in cash due to member payments )
   SET VARIABLE VACCTNUM TO "1110"
   SET VARIABLE VWHAT TO "PD RECV MBR" & .VMEMNUM
   LOAD BAL_JOUR
      .VACCTNUM .VWHAT .VPAYMENT .VTRANDAT P
   END

ENDWHILE

SET MESSAGES ON
SET ERROR MESSAGES ON
RETURN
*(* END OF PUTPAY.CMD )

$COMMAND
CRED_BUY
```

```
*(*********************************************************************
PROGRAM:       CRED_BUY.CMD
AUTHOR:        J. M. GRAHAM
DATE:          JAN 1987     VER. 1.9
DESCRIPTION:   THIS MODULE ALLOWS THE MANAGER TO ENTER ALL PURCHASES
               MADE ON CREDIT TO EITHER A NEW SUPPLIER/VENDOR OR AN
               EXISTING ACCOUNT.  THE MANAGER MUST PROVIDE THE CORRECT
               ACCOUNT NUMBER FOR INVENTORY/ASSET/EXPENSE ACCOUNT
               INCREASES.  ALL INTEREST CHARGED TO AN ACCOUNT MUST BE
               ENTERED LABELING PRODUCT AS INTEREST AND INVOICE NUMBER
               AS ZERO.

TABLES USED:   BAL_JOUR, EXPENSE, ACCT_PAY, AP_CHG
FORMS USED:    CRED_PUR                                          (08)
*********************************************************************)

SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

  *(* ASK USER IF PREVIOUS/ACTIVE ACCOUNT *)
LABEL AGAIN
NEWPAGE
WRITE "ENTERING CREDIT PURCHASES AND CHARGED INTEREST" AT 2,15
WRITE "Is this entry to be posted to an active account? (Y/N)" AT 4,5
FILLIN YOURANS USING " " AT 4,62
IF YOURANS = Y THEN
   GOTO OLDCRED
ENDIF

LABEL NEWCRED *(* Draw entry form to get new cred data )
NEWPAGE       *(* and cred purchase)
DRAW CRED_PUR
WRITE "PRESS PGDN KEY TO ENTER DATA - PRESS ESC KEY TO QUIT" AT 24,8
ENTER VARIABLE VVENDNO VVL:NAME VVF:NAME VVEN:MI VVENADDR VCITY +
      VSTATE VZIP VPHONE VPRODUCT VP:DATE VT:PRICE VINV:NO VACCTNUM +
      VEXP:NO RETURN PGDN ESC
IF #RETURN = ESC THEN
   GOTO ENDCRED
ENDIF

IF VACCTNUM EXISTS THEN   *(* posting inventory increases)
   SET VARIABLE VTX:CODE TO "P"
   LOAD BAL_JOUR
        .VACCTNUM .VPRODUCT .VT:PRICE .VP:DATE .VTX:CODE
   END
ENDIF

IF VEXP:NO EXISTS THEN  *(* posting expense increases)
   SET VARIABLE VTX:CODE TO "P"
   LOAD EXPENSE
        .VEXP:NO .VPRODUCT .VT:PRICE .VP:DATE  .VTX:CODE
   END
ENDIF

   SET VARIABLE VCURBAL TO "0.00"    *(* set initial valve of new cred)
   SET VARIABLE VPOSTDAY TO .#DATE   *(* account's owed balance       )

LOAD ACCT_PAY *(* enter new creditor's data -- open an account)
  .VVENDNO .VVL:NAME .VVF:NAME .VVEN:MI .VVENADDR .VCITY .VSTATE .VZIP +
  .VPHONE .VCURBAL .VPOSTDAY
END
   LOAD AP_CHG  *(* posting credit purchases)
        .VVENDNO .VPRODUCT .VP:DATE .VT:PRICE .VINV:NO
   END

   CLEAR VACCTNUM
   CLEAR VEXP:NO
   GOTO REQAGAIN
```

```
LABEL OLDCRED
NEWPAGE
WRITE "PROCESSING A NEW CHARGE TO A CURRENT ACCT. - ESC TO QUIT" AT 2,5

LABEL REDO
WRITE"PLEASE, ENTER EITHER THE CREDITOR'S ACCOUNT NUMBER OR LAST NAME"
   AT 5,5
WRITE "ONLY ENTER ONE OR THE OTHER!" AT 6,15
FILLIN VVENDNO USING "CREDITOR'S ACCOUNT NUMBER IS - " AT 8,5
FILLIN VVL:NAME USING "CREDITOR'S LAST NAME IS - " AT 10,5
IF VVENDNO FAILS THEN
   *(* search record by creditor's last name to provide data)
   SET POINTER #1 STATUS1 FOR ACCT_PAY WHERE VNL:NAME = .VVL:NAME
   *(* warning that customer not currently on database)
   IF STATUS1 <> 0 THEN
      WRITE "NO SUCH ACCOUNTS PAYABLE ACCOUNT" AT 15,20
      WRITE "PRESS ANY KEY TO REENTER THE CREDITOR'S DATA" AT 18,15
      PAUSE
      GOTO OLDCRED
   ELSE
      SET VARIABLE VVENDNO TO VENDNO IN #1
      GOTO PRTDATA
   ENDIF
ELSE
   *(* search record by creditor's account number )
   SET POINTER #1 STATUS1 FOR ACCT_PAY WHERE VENDNO = .VVENDNO
   *(* CUSTOMER NOT CURRENTLY ON DATABASE *)
   IF STATUS1 <> 0 THEN
      WRITE "NO SUCH ACCOUNTS PAYABLE ACCOUNT" AT 15,20
      WRITE "PRESS ANY KEY TO REENTER THE CREDITOR'S DATA" AT 18,15
      PAUSE
      GOTO OLDCRED
   ELSE
      SET VARIABLE VVL:NAME TO VNL:NAME IN #1
      GOTO PRTDATA
   ENDIF
ENDIF

LABEL PRTDATA
*(* set data from chosen account to printed in form for verification)
   SET VARIABLE VVF:NAME TO VNF:NAME IN #1
   SET VARIABLE VVEN:MI TO VEN:MI IN #1
   SET VARIABLE VVENADDR TO VEN_STRT IN #1
   SET VARIABLE VCITY TO CITY IN #1
   SET VARIABLE VSTATE TO STATE IN #1
   SET VARIABLE VZIP TO ZIPCODE IN #1
   SET VARIABLE VPHONE TO PHONE IN #1
   NEWPAGE
   DRAW CRED_PUR WITH VARIABLE VVENDNO VVF:NAME VVEN:MI VVL:NAME +
        VVENDADDR VCITY VSTATE VZIP VPHONE
 WRITE "PRESS PGDN KEY TO ENTER DATA - PRESS ESC KEY TO QUIT" AT 24,8
   ENTER VARIABLE VP:DATE VT:PRICE VPRODUCT VINV:NO VACCTNUM +
         VEXP:NO RETURN PGDN ESC
   IF #RETURN = ESC THEN
      GOTO ENDCRED
   ENDIF

   IF VACCTNUM EXISTS THEN   *(* posting inventory increases)
      SET VARIABLE VTX:CODE TO "P"
      LOAD BAL_JOUR
           .VACCTNUM .VPRODUCT .VT:PRICE .VP:DATE .VTX:CODE
      END
   ENDIF

   IF VEXP:NO EXISTS THEN    *(* posting expense increases)
      SET VARIABLE VTX:CODE TO "P"
      LOAD EXPENSE
           .VEXP:NO .VPRODUCT .VT:PRICE .VP:DATE .VTX:CODE
      END
   ENDIF

   LOAD AP_CHG   *(* posting credit purchases or charged interest)
       .VVENDNO .VPRODUCT .VP:DATE .VT:PRICE .VINV:NO
   END

   CLEAR VACCTNUM
   CLEAR VEXP:NO

LABEL REQAGAIN  *(* gives user opportunity to make more entries)
   NEWPAGE         *(* without going thru the main menu)
   WRITE "DO YOU HAVE ANOTHER CREDIT PURCHASE TO ENTER? (Y/N)" AT 5,5
   FILLIN YOURANS USING " " AT 5,60
   IF YOURANS = Y THEN
      GOTO AGAIN
   ENDIF

LABEL ENDCRED  *(* return to financial transaction main menu )
   SET MESSAGES ON
   SET ERROR MESSAGES ON
   RETURN
*(* END OF CRED_BUY.CMD *)

$COMMAND
CASH_BUY
*(*******************************************************************
PROGRAM:       CASH_BUY.CMD
AUTHOR:        J. M. GRAHAM
DATE:          DEC 1986    VER. 1.4
DESCRIPTION:   THIS MODULE PROVIDES THE MANAGER WITH A METHOD TO POST
               CHECK(CASH) PURCHASES OF THE CLUB.  CASH ACCOUNT IS
               DECREASED, INVENTORY INCREASED, AND EXPENSE ACCOUNT
               INCREASED.  MANAGER MUST ENTER THE CORRECT ACCOUNT NO.
               TO BE CREDITED/DEBTED.

TABLES USED:   BAL_JOUR, EXPENSE
FORMS USED:    NONE                                            (08)
*******************************************************************)

SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

NEWPAGE
WRITE "CASH(CHECK) PURCHASE BY MONTEREY NAVY FLYING CLUB" AT 8,15
WRITE "THIS IS NOT A CREDIT PURCHASE"  AT 10,20
WRITE "PRESS ANY KEY TO ENTER YOUR CASH PURCHASE"  AT 14,15
PAUSE

LABEL CASHBUY
NEWPAGE
WRITE "CASH(CHECK) PURCHASE ENTRY"  AT 2,25
FILLIN VTX:DATE USING "PURCHASE DATE WAS - (MM/DD/YY): " AT 5,5
FILLIN VAMOUNT USING "AMOUNT SPENT ON PURCHASE WAS - $ "  AT 7,5
FILLIN VITEM USING "ITEM PURCHASED WAS - (12 Character limit): "AT 9,5
FILLIN VBALACCT USING"INCREASE TO ASSET/INVENTORY ACCT NUMBER - "AT11,5
FILLIN VEXPACCT USING "INCREASE TO EXPENSE ACCT NUMBER - " AT 13,5
   WRITE "IS THE ABOVE DATA CORRECT? (Y/N) "  AT 15,20
   FILLIN YOURANS USING " " AT 15,55
   IF YOURANS = N THEN
      CLEAR VAMOUNT
      CLEAR VITEM
      CLEAR VBALACCT
      CLEAR VEXPACCT
      GOTO CASHBUY
   ENDIF

   SET VARIABLE VCASH TO "1110"
   SET VARIABLE VTX:CODE TO "M"
   SET VARIABLE VNAME1 TO "BOUGHT"     *(*  label of item purchased)
   SET VARIABLE VWHAT TO .VNAME1 & .VITEM

   LOAD BAL_JOUR    *(* enter cash deduction to cover purchase)
      .VCASH .VWHAT .VAMOUNT .VTX:DATE .VTX:CODE
   END

IF VEXPACCT EXISTS THEN   *(* posting expense increase due to purchase)
      SET VARIABLE VTX:CODE TO "P"
      LOAD EXPENSE
         .VEXPACCT .VITEM .VAMOUNT .VTX:DATE .VTX:CODE
      END
   ENDIF

IF VBALACCT EXISTS THEN*(* posting inventory increase due to purchase)
      SET VARIABLE VTX:CODE TO "P"
      LOAD BAL_JOUR
         .VBALACCT .VITEM .VAMOUNT .VTX:DATE .VTX:CODE
      END
   ENDIF

   CLEAR VAMOUNT
   CLEAR VITEM
   CLEAR VEXPACCT
   CLEAR VBALACCT
   CLEAR VWHAT
   NEWPAGE
   WRITE "DESIRE TO ENTER ANOTHER CASH PURCHASE?(Y/N)" AT 8,10
   FILLIN YOURANS USING " " AT 8,55
   IF YOURANS = Y THEN
      GOTO CASHBUY
   ENDIF

SET MESSAGES ON
SET ERROR MESSAGES ON
RETURN
*(* END OF CASH_BUY.CMD *)

$COMMAND
PD_ACCT
*(*****************************************************************
PROGRAM:      PD_ACCT.CMD
AUTHOR:       J. M. GRAHAM
DATE:         DEC 1986     VER. 1.5
DESCRIPTION:  THIS MODULE ALLOWS THE MANAGER TO ENTER PAYMENTS UPON
              ACCOUNTS PAYABLE ACCOUNTS.  MANAGER MUST PROVIDE THE
              CREDITOR'S ACCOUNT NUMBER OR NAME, PLUS THE PAYMENT
              DATA.

TABLES USED:  BAL_JOUR, AP_PAID, ACCT_PAY
FORMS USED:   CRED_PAY                                      (08)
*****************************************************************)

SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

LABEL POSTPAY
  CLEAR VVENDNO
  CLEAR VVL:NAME
  CLEAR VPAIDAMT
  CLEAR VWHAT
  CLEAR VCK:NO
NEWPAGE
WRITE "POSTING PAYMENT TO AN ACCOUNT" AT 5,25

WRITE "PLEASE, ENTER EITHER THE CREDITOR'S ACCOUNT NUMBER OR LAST NAME"
    AT 8,5
```

```
WRITE "ENTER ONLY ONE OR THE OTHER!" AT 10,15
FILLIN VVENDNO USING "CREDITOR'S ACCOUNT NUMBER IS - " AT 12,5
FILLIN VVL:NAME USING "CREDITOR'S LAST NAME IS - " AT 14,5
IF VVENDNO EXISTS AND VVL:NAME EXISTS THEN
  WRITE "PLEASE, ENTER EITHER THE CREDITOR'S ACCOUNT NUMBER OR NAME"
      AT 16,10
  WRITE "ENTER ONLY ONE"  AT 17,35
  WRITE "PRESS ANY KEY TO REENTER CREDITOR'S DATA"  AT 19,15
  PAUSE
  GOTO POSTPAY
ENDIF

IF VVENDNO FAILS THEN
   *(* search record by creditor's last name to provide data)
   SET POINTER #1 STATUS1 FOR ACCT_PAY WHERE VNL:NAME = .VVL:NAME
   *(* warning that customer not currently on database)
   IF STATUS1 <> 0 THEN
      WRITE "NO SUCH ACCOUNTS PAYABLE ACCOUNT" AT 16,20
      WRITE "PRESS ANY KEY TO REENTER THE CREDITOR'S DATA" AT 18,15
      PAUSE
      GOTO POSTPAY
   ELSE
      SET VARIABLE VVENDNO TO VENDNO IN #1
      GOTO POSTDATA
   ENDIF
ELSE
   *(* search record by creditor's account number )
   SET POINTER #1 STATUS1 FOR ACCT_PAY WHERE VENDNO = .VVENDNO
   *(* CUSTOMER NOT CURRENTLY ON DATABASE *)
   IF STATUS1 <> 0 THEN
      WRITE "NO SUCH ACCOUNTS PAYABLE ACCOUNT" AT 16,20
      WRITE "PRESS ANY KEY TO REENTER THE CREDITOR'S DATA" AT 18,15
      PAUSE
      GOTO POSTPAY
   ELSE
      SET VARIABLE VVL:NAME TO VNL:NAME IN #1
      GOTO POSTDATA
   ENDIF
ENDIF

LABEL POSTDATA
*(* set data from chosen acct to be printed in form for verification)
   SET VARIABLE VVF:NAME TO VNF:NAME IN #1
   SET VARIABLE VVEN:MI TO VEN:MI IN #1
   SET VARIABLE VVENADDR TO VEN_STRT IN #1
   SET VARIABLE VCITY TO CITY IN #1
   SET VARIABLE VSTATE TO STATE IN #1
   SET VARIABLE VZIP TO ZIPCODE IN #1
   SET VARIABLE VBALOWED TO CURRBAL IN #1
   SET VARIABLE VPOSTDAY TO POSTDATE IN #1

   NEWPAGE
   DRAW CRED_PAY WITH VARIABLE VVENDNO VVF:NAME VVEN:MI VVL:NAME +
        VVENADDR VCITY VSTATE VZIP VBALOWED VPOSTDAY
  WRITE "PRESS PGDN KEY TO ENTER DATA - PRESS ESC KEY TO QUIT" AT 24,8
   ENTER VARIABLE VPAIDAMT VTX:DATE VCK:NO RETURN PGDN ESC
   IF #RETURN = ESC THEN
      GOTO CREDEND
   ENDIF

   SET VARIABLE VCASH TO "1110"
   SET VARIABLE VTX:CODE TO "M"
   SET VARIABLE VWHAT TO "PD ON ACCT" & .VVENDNO

   LOAD BAL_JOUR
        .VCASH .VWHAT .VPAIDAMT .VTX:DATE .VTX:CODE
   END

   LOAD AP_PAID   *(*  posting of payment )
```

```
        .VVENDNO .VTX:DATE .VPAIDAMT .VCK:NO
   END


LABEL REQAGAIN  *(* gives user opportunity to make more entries)
   NEWPAGE        *(* without going thru the main menu)
   WRITE "DO YOU HAVE ANOTHER PAYMENT TO ENTER? (Y/N)" AT 8,15
   FILLIN YOURANS USING " " AT 8,60
   IF YOURANS = Y THEN
      GOTO POSTPAY
   ENDIF

LABEL CREDEND  *(* return to financial transaction main menu )
   SET MESSAGES ON
   SET ERROR MESSAGES ON
   RETURN
*(* END OF PD_ACCT.CMD *)


$COMMAND
EDIT_CHG
*(*****************************************************************
PROGRAM:       EDIT_CHG.CMD
AUTHOR:        J. M. GRAHAM
DATE:          JAN 1987      VER.  1.4
DESCRIPTION:   THIS MODULE ALLOWS EDITING OR DELETING OF A MEMBER
               CHARGE TO HIS ACCOUNT THAT MAY HAVE BEEN ENTERED IN
               ERROR.  MANAGER MUST KNOW MEMBER NUMBER, DATE OF
               CHARGE, INVOICE/CHARGE SHEET NUMBER TO ENTER THE
               CHANGE.  MANAGER MUST INDICATE WHETHER THE ERROR WAS
               DISCOVERED PROIR TO OR AFTER THE END-OF-THE-MONTH
               BILLING.  THE MODULE HANDLES THE TRANSACTION DIFFERENTLY
               BASED UPON THE TIME OF ERROR DISCOVERY.

TABLES USED:   MEM_REC, MEM_CHG, MEM_FLT, INST_REC, A/C_REC, INCOME,
               A/C_HRS, FLT_REC
FORMS USED:    ADJMBRFM, ADJCFIFM, ADJINCFM, ADJEXPFM, FLTHRSFM
               ADJMBRFL, ADJFLTHR                                 (08)
*****************************************************************)

SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

LABEL BEGINEDIT
NEWPAGE
WRITE "PROCESSING CORRECTIONS OR DELETIONS OF MEMBER CHARGES"  AT 2,15
WRITE "-----------------------------------------------------"  AT 3,15
WRITE "Corrections are handled differently if discovered prior to"
    AT 6,10
WRITE "or after the regular end-of-month billing."  AT 7,5
WRITE "ENTER:"  AT 9,5
WRITE "1 - TO HANDLE ERRORS DISCOVERED PRIOR TO END-OF-MONTH BILLING"
     AT 10,5
WRITE "2 - TO HANDLE ERRORS DISCOVERED AFTER END-OF-MONTH BILLING"
     AT 12,5
WRITE "3 - EXIT WITHOUT AN UPDATE"  AT 14,5
WRITE "CHOICE: "  AT 16,5
FILLIN YOURCH USING " " AT 16,15
IF YOURCH = 1 THEN
   GOTO PROIREOM
ENDIF
IF YOURCH = 2 THEN
   GOTO AFTEREOM
ENDIF
IF YOURCH = 3 THEN
   GOTO EDITEND
ENDIF
```

```
LABEL PROIREOM
NEWPAGE
WRITE "YOU ARE ABOUT TO EDIT OR DELETE A MEMBER'S CHARGE" AT 2,10
WRITE "FOLLOW THE PROMPTS AT THE TOP OF YOUR EDIT SCREEN" AT 4,10
WRITE "TO EFFECT THE EDIT/DELETION AS YOU DESIRE." AT 5,10
WRITE "AFTER AN EDIT IS MADE YOU MUST SAVE THE EDIT BY SELECTING"
    AT 6,10
WRITE "OR HIGHLIGHTING THE 'CHANGE' OPTION AND PRESS THE ENTER KEY"
    AT 7,10

LABEL GETNUM
WRITE "YOUR MUST KNOW THE MEMBER'S NUMBER" AT 9,20
FILLIN VMEMNUM USING "MEMBER'S NUMBER IS - "  AT 11,10
FILLIN VINV:NO USING "CHARGE SHEET/INVOICE NUMBER (IF KNOWN) - "
    AT 13,10

SET POINTER #1 STATUS1 FOR MEM_REC WHERE MEM_NUM EQ .VMEMNUM
IF STATUS1 <> 0 THEN
   WRITE "NO SUCH MEMBER ACCOUNT"  AT 15,20
   WRITE "PRESS ANY KEY TO REENTER THE MEMBER'S NUMBER"  AT 16,15
   PAUSE
   CLEAR VMEMNUM
   CLEAR VINV:NO
   NEWPAGE
   GOTO GETNUM
ENDIF

   SET VARIABLE NAME1 TO F:NAME IN #1
   SET VARIABLE NAME2 TO MID_INT IN #1
   SET VARIABLE NAME3 TO .NAME1 & .NAME2
   SET VARIABLE NAME4 TO L:NAME IN #1
   SET VARIABLE VFULNAM TO .NAME3 & .NAME4
   SET VARIABLE VSTREET TO STREET IN #1
   SET VARIABLE VCITY TO CITY IN #1
   SET VARIABLE VSTATE TO STATE IN #1
   SET VARIABLE VADDR TO .VCITY & .VSTATE
   SET VARIABLE VZIP TO ZIPCODE IN #1

   WRITE "MEMBER: " AT 15,20
   WRITE .VFULNAM  AT 15,28
   WRITE .VSTREET  AT 16,28
   WRITE .VADDR AT 17,28
   WRITE .VZIP AT 18,28
   WRITE "IS THIS THE CORRECT ACCOUNT TO BE ADJUSTED?(Y/N)"  AT 20,15
   FILLIN YOURANS USING " "  AT 20,70
   IF YOURANS = N THEN
      CLEAR VMEMNUM
      CLEAR VINV:NO
      NEWPAGE
      GOTO GETNUM
   ENDIF

IF VINV:NO EXITS THEN
 EDIT USING ADJMBRFM WHERE MEM_NUM EQ .VMEMNUM AND INVO_NUM EQ .VINV:NO
ELSE
 EDIT USING ADJMEMFM SORTED BY MEM_NUM WHERE MEM_NUM EQ .VMEMNUM
ENDIF

LABEL FLTMENU
NEWPAGE
WRITE "ADDITIONAL ADJUSTMENTS THAT YOUR CHANGE MAY REQUIRE"  AT 3,10
WRITE "---------------------------------------------------"  AT 4,10
WRITE "ENTER:"  AT 6,5
WRITE "1 - FLIGHT HOURS ON AIRCRAFT AND MEMBER CHANGE"  AT 8,5
WRITE "2 - INSTRUCTOR FLIGHT HOURS REQUIRE ADJUSTMENT"  AT 10,5
WRITE"3 - INCOME ACCOUNT ADJUSTMENT DUE TO CHARGE AMOUNT CHANGE"AT 12,5
WRITE"4 - EXPENSE ACCOUNT ADJUSTMENT DUE TO CHARGE AMOUNT CHANGE"AT14,5
WRITE"5 - QUIT ... RETURN TO FINANCIAL MENU"  AT 16,5
WRITE "CHOICE:"  AT 18,5
```

```
FILLIN YOURCH USING " "  AT 18,15
IF YOURCH = 1 THEN
 FILLIN VACNUM USING "CHANGE HOURS ON AIRCRAFT NUMBER - "  AT 20,20
 EDIT USING FLTHRSFM WHERE A/C_NUM EQ .VACNUM
 EDIT USING ADJMBRFL WHERE MEM_NUM EQ .VMEMNUM AND LAST_A/C EQ .VACNUM
 EDIT USING ADJFLTHR WHERE A/C_NUM EQ .VACNUM
 CLEAR VACNUM
 GOTO FLTMENU
ENDIF
IF YOURCH = 2 THEN
 FILLIN VINSTNUM USING "CREDIT INSTRUCTOR NUMBER - "  AT 20,20
 EDIT USING ADJCFIFM WHERE INST_NUM EQ .VINSTNUM AND MEM_NUM EQ .VMEMNUM
 CLEAR VINSTNUM
 GOTO FLTMENU
ENDIF
IF YOURCH = 3 THEN
  FILLIN VACCTNUM USING "CHANGE REQUIRED TO INCOME ACCOUNT NUMBER - "
     AT  2,10
  EDIT USING ADJINCFM WHERE ACCT_NUM EQ .VACCTNUM
  CLEAR VACCTNUM
  GOTO FLTMENU
ENDIF
IF YOURCH = 4 THEN
  FILLIN VACCTNUM USING "CHANGE REQUIRED TO EXPENSE ACCOUNT NUMBER - "
      AT 20,10
  EDIT USING UDJEXPFM WHERE ACCT_NUM EQ .VACCTNUM
  CLEAR VACCTNUM
  GOTO FLTMENU
ENDIF
IF YOURCH = 5 THEN
  CLEAR VMEMNUM
  CLEAR VINV:NO
  GOTO EDITEND
ENDIF
CLEAR VMEMNUM
CLEAR VINV:NO
GOTO REQAGAIN

LABEL AFTEREOM
NEWPAGE
WRITE "CORRECTIONS TO CHARGE AMOUNTS MADE TO MEMBER'S ACCOUNTS"  AT 2,5
WRITE "-------------------------------------------------------"  AT 3,5
WRITE "NOTE:  SINCE CHANGES ARE REQUIRED AFTER NORMAL BILLING,"  AT 5,5
WRITE "ANY CHANGES TO AN EXPENSE ACCOUNTS MUST BE MADE UNDER" AT 6,12
WRITE "THE MANAGER'S UPDATE ENTRY SELECTION OF THE FINANCIAL MENU"
    AT 7,12
WRITE "MEMBER FLIGHT HOUR CHANGES CAN NOT BE MADE AT THIS TIME."AT 8,12
WRITE"TOTAL FLIGHT HOURS FOR A SPECIFIC AIRCRAFT ARE MADE UNDER"AT 9,12
WRITE "A/C MAINTENANCE/FLT HOUR UPDATE SELECTION OF THE MAIN MENU"
     AT 10,12

WRITE "ENTER:"  AT 12,5
WRITE "1 - TO CORRECT A CHARGE AMOUNT ENTRY ERROR"  AT 14,5
WRITE "2 - TO CORRECT CHARGE MADE TO THE WRONG ACCOUNT"  AT 16,5
WRITE "3 - BOTH CHARGE AMOUNT AND MEMBER ACCOUNT POSTED ARE WRONG"
     AT 18,5
WRITE "4 - TO EXIT WITH NO UPDATE"  AT 20,5
WRITE "CHOICE:"  AT 22,5
FILLIN YOURCH USING " " AT 22,15
IF YOURCH = 1 THEN
   *(-- corrections due to incorrect charge entry)
   GOTO GETACCT
ENDIF
IF YOURCH = 2 THEN
   *(--  corrections due to charge posted to incorrect MBR account)
   GOTO GETFIRST
ENDIF
IF YOURCH = 3 THEN
   *(--corrections due to posting an incorrect charge to a wrong acct)
   GOTO GETBOTH
ENDIF
IF YOURCH = 4 THEN
   *(--  no corretions necessary; return to financial edit menu )
   GOTO EDITEND
ENDIF

LABEL GETACCT
NEWPAGE
   FILLIN VMEMNUM USING "MEMBER'S NUMBER IS - " AT 5,20

   SET POINTER #1 STATUS1 FOR MEM_REC WHERE MEM_NUM EQ .VMEMNUM
   IF STATUS1 <> 0 THEN
     WRITE "NO SUCH MEMBER ACCOUNT"  AT 15,20
     WRITE "PRESS ANY KEY TO REENTER THE MEMBER'S NUMBER"  AT 16,15
     PAUSE
     CLEAR VMEMNUM
     GOTO GETACCT
   ENDIF

   SET VARIABLE NAME1 TO F:NAME IN #1
   SET VARIABLE NAME2 TO MID_INT IN #1
   SET VARIABLE NAME3 TO .NAME1 & .NAME2
   SET VARIABLE NAME4 TO L:NAME IN #1
   SET VARIABLE VFULNAM TO .NAME3 & .NAME4
   SET VARIABLE VSTREET TO STREET IN #1
   SET VARIABLE VCITY TO CITY IN #1
   SET VARIABLE VSTATE TO VSTATE IN #1
   SET VARIABLE VADDR TO .VCITY & .VSTATE
   SET VARIABLE VZIP TO ZIPCODE IN #1

   WRITE "MEMBER: " AT 15,20
   WRITE .VFULNAM AT 15,28
   WRITE .VSTREET AT 16,28
   WRITE .VADDR AT 17,28
   WRITE .VZIP AT 18,28
   WRITE "IS THIS THE CORRECT ACCOUNT TO BE ADJUSTED?(Y/N)"  AT 20,15
   FILLIN YOURANS USING " "  AT 20,70
   IF YOURANS = N THEN
      CLEAR VMEMNUM
      GOTO GETACCT
   ENDIF

 NEWPAGE
 FILLIN VBADAMT USING "CHARGE AMOUNT ENTERED IN ERROR WAS - " AT 7,5
 FILLIN VAMT USING "CORRECT CHARGE AMOUNT IS - " AT 9,5
 FILLIN VINVNO USING "INVOICE NUMBER OF CHARGE WAS -" AT 11,5
 FILLIN VACCTNUM USING "INCOME ACCOUNT EFFECTED BY CHANGE IS - "AT 13,5
   SET VARIABLE VTX:DATE TO .#DATE
   SET VARIABLE VDESCRP TO "CHG COR" & .VMEMNUN
   SET VARIABLE VDESCRP1 TO "CHARGE CORRECTION"
   SET VARIABLE VCK:NO TO "0"
   SET VARIABLE VHRS TO "0"
   SET VARIABLE VA/CNUM TO "NONE"
   LOAD INCOME  *(-- correct income posted in error.)
      .VACCTNUM .VDESCRP .VBADAMT .VTX:DATE M
   END
   LOAD INCOME *(-- show correct income earned from member )
      .VACCTNUM .VDESCRP1 .VAMT .VTX:DATE P
   END
   LOAD MEM_PAY *(-- corrective charge to zero incorrect charge entry)
      .VMEMNUM .VCK:NO .VTX:DATE .VBADAMT .VDESCRP1
   END
   LOAD MEM_CHG  *(-- show correct member's charge amount)
      .VMEMNUM .VINVNO .VTX:DATE .VAMT .VDESCRP .VHRS .VA/CNUM
   END
   CLEAR VBADAMT
   CLEAR VAMT
   CLEAR VACCTNUM
   CLEAR VINVNO
   CLEAR VMEMNUN
   GOTO REQAGAIN

LABEL GETFIRST
NEWPAGE
FILLIN VBADMBR USING"CHARGE POSTED IN ERROR TO ACCOUNT NUMBER - "
    AT 10,10
  SET POINTER #1 STATUS1 FOR MEM_REC WHERE MEM_NUM EQ .VBADMBR
  IF STATUS1 <> 0 THEN
    WRITE "NO SUCH MEMBER ACCOUNT"  AT 15,20
    WRITE "PRESS ANY KEY TO REENTER THE MEMBER'S NUMBER"  AT 16,15
    PAUSE
    CLEAR VBADMBR
    GOTO GETFIRST
  ENDIF

  SET VARIABLE NAME1 TO F:NAME IN #1
  SET VARIABLE NAME2 TO MID_INT IN #1
  SET VARIABLE NAME3 TO .NAME1 & .NAME2
  SET VARIABLE NAME4 TO L:NAME IN #1
  SET VARIABLE VFULNAM TO .NAME3 & .NAME4
  SET VARIABLE VSTREET TO STREET IN #1
  SET VARIABLE VCITY TO CITY IN #1
  SET VARIABLE VSTATE TO STATE IN #1
  SET VARIABLE VADDR TO .VCITY & .VSTATE
  SET VARIABLE VZIP TO ZIPCODE IN #1

  WRITE "CHARGE POSTED IN ERROR TO: "  AT 15,10
  WRITE .VFULNAM AT 15,40
  WRITE .VSTREET AT 16,40
  WRITE .VADDR AT 17,40
  WRITE .VZIP AT 18,40
  WRITE "IS THIS THE ACCOUNT CHARGE WAS POSTED IN ERROR TO?(Y/N)"
      AT 20,10
  FILLIN YOURANS USING " " AT 20,68
  IF YOURANS = N THEN
     CLEAR VBADMBR
     GOTO GETFIRST
  ENDIF

LABEL GETSECOND
NEWPAGE
  FILLIN VMEMNUM USING "CHARGE SHOULD HAVE BEEN MADE TO ACCOUNT - "
      AT 12,10

  SET POINTER #2 STATUS2 FOR MEM_REC WHERE MEM_NUM EQ .VMEMNUM
  IF STATUS2 <> 0 THEN
     WRITE "NO SUCH MEMBER ACCOUNT"  AT 15,20
     WRITE "PRESS ANY KEY TO REENTER THE MEMBER'S NUMBER"  AT 16,15
     PAUSE
     CLEAR VMEMNUM
     GOTO GETSECOND
  ENDIF

  SET VARIABLE NAME1 TO F:NAME IN #2
  SET VARIABLE NAME2 TO MID_INT IN #2
  SET VARIABLE NAME3 TO .NAME1 & .NAME2
  SET VARIABLE NAME4 TO L:NAME IN #2
  SET VARIABLE VFULNAM TO .NAME3 & .NAME4
  SET VARIABLE VSTREET TO STREET IN #2
  SET VARIABLE VCITY TO CITY IN #2
  SET VARIABLE VSTATE TO STATE IN #2
  SET VARIABLE VADDR TO .VCITY & .VSTATE
  SET VARIABLE VZIP TO ZIPCODE IN #2

  WRITE "CHARGE SHOULD BE POSTED TO: "  AT 6,10
  WRITE .VFULNAM AT 6,40
  WRITE .VSTREET AT 7,40
  WRITE .VADDR AT 8,40
  WRITE .VZIP AT 9,40
  WRITE "IS THIS THE CORRECT MEMBER'S ACCOUNT TO BE CHARGED?(Y/N)"
       AT 11,5
  FILLIN YOURANS USING " " AT 11,72
  IF YOURANS = N THEN
     CLEAR VMEMNUM
     GOTO GETSECOND
  ENDIF

  FILLIN VAMT USING "AMOUNT OF CHARGE TO BE POSTED - "  AT 14,10
  FILLIN VINVNO USING "INVOICE NUMBER OF CHARGE WAS - "  AT 18,10
  SET VARIABLE VCK:NO TO "0"
  SET VARIABLE VTX:DATE TO .#DATE
  SET VARIABLE VDESCRP TO "CHARGE CORRECTION"
  SET VHRS TO "0"
  SET VA/CNUM TO "NONE"

  LOAD MEM_CHG  *(* post correct charge to account)
      .VMEMNUM .VINVNO .VTX:DATE .VAMT .VDESCRP .VHRS .VA/CNUM
  END
  LOAD MEM_PAY  *(* zero incorrect charge to account)
      .VBADMBR .VCK:NO .VTX:DATE .VAMT .VDESCRP
  END
  CLEAR VMEMNUM
  CLEAR VBADMBR
  CLEAR VAMT
  CLEAR VINVNO
  GOTO REQAGAIN

LABEL GETBOTH
NEWPAGE
  FILLIN VBADMBR USING "CHARGE POSTED IN ERROR TO ACCOUNT NUMBER - "
        AT  10,10
  SET POINTER #1 STATUS1 FOR MEM_REC WHERE MEM_NUM EQ .VBADMBR
  IF STATUS1 <> 0 THEN
     WRITE "NO SUCH MEMBER ACCOUNT"  AT 15,20
     WRITE "PRESS ANY KEY TO REENTER THE MEMBER'S NUMBER"  AT 16,15
     PAUSE
     CLEAR VBADMBR
     GOTO GETBOTH
  ENDIF

  SET VARIABLE NAME1 TO F:NAME IN #1
  SET VARIABLE NAME2 TO MID_INT IN #1
  SET VARIABLE NAME3 TO .NAME1 & .NAME2
  SET VARIABLE NAME4 TO L:NAME IN #1
  SET VARIABLE VFULNAM TO .NAME3 & .NAME4
  SET VARIABLE VSTREET TO STREET IN #1
  SET VARIABLE VCITY TO CITY IN #1
  SET VARIABLE VSTATE TO STATE IN #1
  SET VARIABLE VADDR TO .VCITY & .VSTATE
  SET VARIABLE VZIP TO ZIPCODE IN #1

  WRITE "CHARGE POSTED IN ERROR TO:"  AT 15,10
  WRITE .VFULNAM AT 15,40
  WRITE .VSTREET AT 16,40
  WRITE .VADDR AT 17,40
  WRITE .VZIP AT 18,40
  WRITE "IS THIS THE ACCOUNT THE CHARGE WAS POSTED IN ERROR TO?(Y/N)"
        AT  20,5
  FILLIN YOURANS USING " "  AT 20,70
  IF YOURANS = N THEN
     CLEAR VBADMBR
     GOTO GETBOTH
  ENDIF

LABEL GETNEXT
NEWPAGE
```

```
  FILLIN VMEMNUM USING "CHARGE SHOULD HAVE BEEN MADE TO ACCOUNT - "
      AT  3,10
  SET POINTER #2 STATUS2 FOR MEM_REC WHERE MEM_NUM EQ .VMEMNUM
  IF STATUS2 <> 0 THEN
     WRITE "NO SUCH MEMBER ACCOUNT"  AT 15,20
     WRITE "PRESS ANY KEY TO REENTER THE MEMBER'S NUMBER"  AT 16,15
     PAUSE
     CLEAR VMEMNUM
     GOTO GETNEXT
  ENDIF

  SET VARIABLE NAME1 TO F:NAME IN #2
  SET VARIABLE NAME2 TO MID_INT IN #2
  SET VARIABLE NAME3 TO .NAME1 & .NAME2
  SET VARIABLE NAME4 TO L:NAME IN #2
  SET VARIABLE VFULNAM TO .NAME3 & .NAME4
  SET VARIABLE VSTREET TO STREET IN #2
  SET VARIABLE VCITY TO CITY IN #2
  SET VARIABLE VSTATE TO STATE IN #2
  SET VARIABLE VADDR TO .VCITY & .VSTATE
  SET VARIABLE VZIP TO ZIPCODE IN #2

  WRITE "CHARGE TO BE POSTED TO: " AT 5,10
  WRITE .VFULNAM AT 5,40
  WRITE .VSTREET AT 6,40
  WRITE .VADDR AT 7,40
  WRITE .VZIP AT 8,40
  WRITE "IS THIS THE CORRECT ACCOUNT TO BE CHARGED?(Y/N)"  AT 10,10
  FILLIN YOURANS USING " "  AT 10,65
  IF YOURANS = N THEN
     CLEAR VMEMNUM
     GOTO GETNEXT
  ENDIF

 FILLIN VBADAMT USING "CHARGE AMOUNT ENTERED IN ERROR WAS - "  AT 12,5
 FILLIN VAMT USING "CORRECT CHARGE AMOUNT IS - "  AT 14,5
 FILLIN VINVNO USING "INVOICE NUMBER OF CHARGE - "  AT 16,5
 FILLIN VACCTNUM USING"NUMBER OF INCOME ACCOUNT EFFECTED IS - "AT 18,5
  SET VARIABLE VTX:DATE TO .#DATE
  SET VARIABLE VDESCRP TO "CHG COR" & .VBADMBR
  SET VARIABLE VDESCRP1 TO "CHARGE CORRECTION"
  SET VARIABLE VCK:NO TO "0"
  SET VARIABLE VHRS TO "0"
  SET VARIABLE VA/CNUM TO "NONE"

  LOAD INCOME  *(-- correct income posted in error )
      .VACCTNUM .VDESCRP .VBADAMT .VTX:DATE M
  END
  LOAD INCOME  *(--  show correct income earned from member )
      .VACCTNUM .VDESCRP1 .VAMT .VTX:DATE P
  END
  LOAD MEM_CHG  *(--  corrective charge to zero the correct account)
      .VMEMNUM .VINVNO .VTX:DATE .VAMT .VDESCRP1 .VHRS .VA/CNUM
  END
  LOAD MEM_PAY  *(--  show correction of incorrect charge to account )
      .VBADMBR .VCK:NO .VTX:DATE .VBADAMT .VDESCRP1
  END
  CLEAR VBADAMT
  CLEAR VAMT
  CLEAR VINVNO
  CLEAR VACCTNUM
  CLEAR VMEMNUM
  CLEAR VBADMBR
  GOTO REQAGAIN

LABEL REQAGAIN
NEWPAGE
WRITE "DO YOU WISH TO EDIT ANOTHER MEMBER CHARGE?(Y/N)" AT 5,10
FILLIN YOURANS USING " " AT 5,60
```

```
IF YOURANS = Y THEN
   GOTO BEGINEDIT
ENDIF

LABEL EDITEND
SET MESSAGES ON
SET ERROR MESSAGES ON
RETURN
*(* END OF EDIT_CHG.CMD )

$COMMAND
EDIT_PAY
*(*********************************************************************
PROGRAM:      EDIT_PAY.CMD
AUTHOR:       J. M. GRAHAM
DATE:         JAN 1987      VER.  1.9
DESCRIPTION:  THIS MODULE ALLOWS EDITING OR DELETING OF A MEMBER
              PAYMENT TO HIS ACCOUNT IF IN ERROR.  MANAGER MUST
              INDICATE WHETHER THE ERROR WAS DISCOVERED PROIR TO
              OR AFTER THE END-OF-MONTH BILLING.  THE MODULE
              HANDLES THE TRANSACTION DIFFERENTLY BASED UPON THE
              TIME OF ERROR DISCOVERY.  THE MANAGER MUST KNOW THE
              MEMBER NUMBER AND ORIGINAL PAYMEMT AMOUNT MADE.

TABLES USED:  MEM_REC, MEM_PAY, BAL_JOUR
FORMS USED:   ADJMEMFM, ADJUSTFM                               (08)
*********************************************************************)

SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

LABEL BEGINEDIT
NEWPAGE
WRITE "PROCESSING CORRECTIONS OR DELETIONS OF MEMBER PAYMENTS"  AT 2,15
WRITE "-----------------------------------------------------"  AT 3,15
WRITE "Corrections are handled differently if discovered prior to"
    AT  6,10
WRITE "or after the regular end-of-month billing."  AT 7,5
WRITE "ENTER:"  AT 9,5
WRITE "1 - TO HANDLE ERRORS DISCOVERED PRIOR TO END-OF-MONTH BILLING"
    AT  10,5
WRITE "2 - TO HANDLE ERRORS DISCOVERED AFTER END-OF-MONTH BILLING"
    AT  12,5
WRITE "3 - EXIT WITHOUT AN UPDATE"  AT 14,5
WRITE "CHOICE: "  AT 16,5
FILLIN YOURCH USING " " AT 16,15
IF YOURCH = 1 THEN
   GOTO PROIREOM
ENDIF
IF YOURCH = 2 THEN
   GOTO AFTEREOM
ENDIF
IF YOURCH = 3 THEN
   GOTO EDITEND
ENDIF

LABEL PROIREOM
NEWPAGE
WRITE "YOU ARE ABOUT TO EDIT OR DELETE A MEMBER'S PAYMENT" AT 2,10
WRITE "FOLLOW THE PROMPTS AT THE TOP OF YOUR EDIT SCREEN" AT 4,10
WRITE "TO EFFECT THE EDIT/DELETION AS YOU DESIRE." AT 5,10
WRITE "AFTER AN EDIT IS MADE YOU MUST SAVE THE EDIT BY SELECTING"
     AT  6,10
WRITE "OR HIGHLIGHTING THE 'CHANGE' OPTION AND PRESS THE ENTER KEY"
     AT  7,10

LABEL GETNUM
WRITE "YOUR MUST KNOW THE MEMBER'S NUMBER" AT 9,20
```

```
FILLIN VMEMNUM USING "MEMBER'S NUMBER IS - "  AT 11,10
FILLIN VCK:NO USING "MEMBER PAID BY CHECK NUMBER (IF KNOWN) - "
    AT 13,10
SET POINTER #1 STATUS1 FOR MEM_REC WHERE MEM_NUM EQ .VMEMNUM
IF STATUS1 <> 0 THEN
   WRITE "NO SUCH MEMBER ACCOUNT"  AT 15,20
   WRITE "PRESS ANY KEY TO REENTER THE MEMBER'S NUMBER"  AT 16,15
   PAUSE
   NEWPAGE
   CLEAR VMEMNUM
   CLEAR VCK:NO
   GOTO GETNUM
ENDIF

   SET VARIABLE NAME1 TO F:NAME IN #1
   SET VARIABLE NAME2 TO MID_INT IN #1
   SET VARIABLE NAME3 TO .NAME1 & .NAME2
   SET VARIABLE NAME4 TO L:NAME IN #1
   SET VARIABLE VFULNAM TO .NAME3 & .NAME4
   SET VARIABLE VSTREET TO STREET IN #1
   SET VARIABLE VCITY TO CITY IN #1
   SET VARIABLE VSTATE TO STATE IN #1
   SET VARIABLE VADDR TO .VCITY & .VSTATE
   SET VARIABLE VZIP TO ZIPCODE IN #1
   SET VARIABLE VCASH TO "1110"
   SET VARIABLE VWHAT TO "PD ON ACCT" & .VMEMNUM

   WRITE "MEMBER: " AT 15,20
   WRITE .VFULNAM  AT 15,28
   WRITE .VSTREET  AT 16,28
   WRITE .VADDR AT 17,28
   WRITE .VZIP AT 18,28
   WRITE "IS THIS THE CORRECT ACCOUNT TO BE ADJUSTED?(Y/N)"  AT 20,15
   FILLIN YOURANS USING " "  AT 20,70
   IF YOURANS = N THEN
      NEWPAGE
      CLEAR VMEMNUM
      CLEAR VCK:NO
      GOTO GETNUM
   ENDIF

IF VCK:NO EXITS THEN
   EDIT USING ADJMEMFM WHERE MEM_NUM EQ .VMEMNUM AND CHK_NUM EQ .VCK:NO
   EDIT USING ADJUSTFM WHERE ACCT_NUM EQ .VCASH AND WHAT EQ .VWHAT
ELSE
   EDIT USING ADJMEMFM SORTED BY MEM_NUM WHERE MEM_NUM EQ .VMEMNUM
   EDIT USING ADJUSTFM WHERE ACCT_NUM EQ .VCASH AND WHAT EQ .VWHAT
ENDIF
WRITE "MAKING CORRECTION TO ACCOUNT"  AT 10,20
CLEAR VMEMNUM
CLEAR VCK:NO
GOTO REQAGAIN

LABEL AFTEREOM
NEWPAGE
WRITE "CORRECTIONS TO PAYMENTS MADE TO MEMBER'S ACCOUNTS"  AT 5,10
WRITE "-------------------------------------------------"  AT 6,10
WRITE "ENTER:"  AT 10,5
WRITE "1 - TO CORRECT A PAYMENT ENTRY ERROR"  AT 12,5
WRITE "2 - TO CORRECT PAYMENT MADE TO THE WRONG ACCOUNT"  AT 14,5
WRITE "3 - BOTH PAYMENT AMOUNT AND MEMBER ACCOUNT POSTED ARE WRONG"
     AT  16,5
WRITE "4 - TO EXIT WITH NO UPDATE"  AT 18,5
WRITE "CHOICE:"  AT 20,5
FILLIN YOURCH USING " " AT 20,15
IF YOURCH = 1 THEN
   *(-- corrections due to incorrect payment entry)
   GOTO GETACCT
ENDIF
```

```
IF YOURCH = 2 THEN
   *(--  corrections due to payment posted to incorrect MBR account)
   GOTO GETFIRST
ENDIF
IF YOURCH = 3 THEN
*(--corrections due to posting an incorrect payment to a wrong acct)
   GOTO GETBOTH
ENDIF
IF YOURCH = 4 THEN
   *(--  no corretions necessary; return to financial edit menu )
   GOTO EDITEND
ENDIF

LABEL GETACCT
NEWPAGE
   FILLIN VMEMNUM USING "MEMBER'S NUMBER IS - " AT 5,20

   SET POINTER #1 STATUS1 FOR MEM_REC WHERE MEM_NUM EQ .VMEMNUM
   IF STATUS1 <> 0 THEN
     WRITE "NO SUCH MEMBER ACCOUNT"  AT 15,20
     WRITE "PRESS ANY KEY TO REENTER THE MEMBER'S NUMBER"  AT 16,15
     PAUSE
     CLEAR VMEMNUM
     GOTO GETACCT
   ENDIF

   SET VARIABLE NAME1 TO F:NAME IN #1
   SET VARIABLE NAME2 TO MID_INT IN #1
   SET VARIABLE NAME3 TO .NAME1 & .NAME2
   SET VARIABLE NAME4 TO L:NAME IN #1
   SET VARIABLE VFULNAM TO .NAME3 & .NAME4
   SET VARIABLE VSTREET TO STREET IN #1
   SET VARIABLE VCITY TO CITY IN #1
   SET VARIABLE VSTATE TO VSTATE IN #1
   SET VARIABLE VADDR TO .VCITY & .VSTATE
   SET VARIABLE VZIP TO ZIPCODE IN #1

   WRITE "MEMBER: " AT 15,20
   WRITE .VFULNAM AT 15,28
   WRITE .VSTREET AT 16,28
   WRITE .VADDR AT 17,28
   WRITE .VZIP AT 18,28
   WRITE "IS THIS THE CORRECT ACCOUNT TO BE ADJUSTED?(Y/N)"  AT 20,15
   FILLIN YOURANS USING " "  AT 20,70
   IF YOURANS = N THEN
      CLEAR VMEMNUM
      GOTO GETACCT
   ENDIF
   NEWPAGE
   FILLIN VBADAMT USING "PAYMENT AMOUNT ENTERED IN ERROR WAS - " AT 7,5
   FILLIN VAMT USING "CORRECT PAYMENT AMOUNT IS - " AT 9,5
   FILLIN VCK:NO USING "CHECK NUMBER OF PAYMENT -" AT 11,5
   SET VARIABLE VTX:DATE TO .#DATE
   SET VARIABLE VDESCRP TO "PD COR" & .VMEMNUN
   SET VARIABLE VDESCRP1 TO "PAYMENT CORRECTION"
   SET VARIABLE VINVNO TO "NONE"
   SET VARIABLE VHRS TO "0"
   SET VARIABLE VA/CNUM TO "NONE"
   SET VARIABLE VCASH TO "1110"
   LOAD BAL_JOUR  *(*  correct cash account by errored amount)
      .VCASH .VDESCRP1 .VBADAMT .VTX:DATE M
   END
   LOAD BAL_JOUR *(*  show correct amount received from member )
      .VCASH .VDESCRP .VAMT .VTX:DATE P
   END
   LOAD MEM_CHG  *(* corrective charge to zero incorrect payment entry)
      .VMEMNUM .VINVNO .VTX:DATE .VBADAMT .VDESCRP1 .VHRS .VA/CNUM
   END
   LOAD MEM_PAY  *(*  show correct member's payment amount)
```

```
       .VMEMNUM .VCK:NO .VTX:DATE .VAMT .VDESCRP1
    END
    CLEAR VMEMNUM
    CLEAR VBADAMT
    CLEAR VAMT
    CLEAR VCK:NO
    GOTO REQAGAIN

LABEL GETFIRST
NEWPAGE
  FILLIN VBADMBR USING "PAYMENT POSTED IN ERROR TO ACCOUNT NUMBER - "
       AT  10,10

  SET POINTER #1 STATUS1 FOR MEM_REC WHERE MEM_NUM EQ .VBADMBR
  IF STATUS1 <> 0 THEN
    WRITE "NO SUCH MEMBER ACCOUNT"  AT 15,20
    WRITE "PRESS ANY KEY TO REENTER THE MEMBER'S NUMBER"  AT 16,15
    PAUSE
    CLEAR VBADMBR
    GOTO GETFIRST
  ENDIF

  SET VARIABLE NAME1 TO F:NAME IN #1
  SET VARIABLE NAME2 TO MID_INT IN #1
  SET VARIABLE NAME3 TO .NAME1 & .NAME2
  SET VARIABLE NAME4 TO L:NAME IN #1
  SET VARIABLE VFULNAM TO .NAME3 & .NAME4
  SET VARIABLE VSTREET TO STREET IN #1
  SET VARIABLE VCITY TO CITY IN #1
  SET VARIABLE VSTATE TO STATE IN #1
  SET VARIABLE VADDR TO .VCITY & .VSTATE
  SET VARIABLE VZIP TO ZIPCODE IN #1

  WRITE "PAYMENT POSTED IN ERROR TO: "  AT 15,10
  WRITE .VFULNAM AT 15,40
  WRITE .VSTREET AT 16,40
  WRITE .VADDR AT 17,40
  WRITE .VZIP AT 18,40
  WRITE "IS THIS THE ACCOUNT PAYMENT WAS POSTED IN ERROR TO?(Y/N)"
        AT  20,10
  FILLIN YOURANS USING " " AT 20,68
  IF YOURANS = N THEN
     CLEAR VBADMBR
     GOTO GETFIRST
  ENDIF

LABEL GETSECOND
NEWPAGE
  FILLIN VMEMNUM USING "PAYMENT SHOULD HAVE BEEN MADE TO ACCOUNT - "
       AT  12,10

  SET POINTER #2 STATUS2 FOR MEM_REC WHERE MEM_NUM EQ .VMEMNUM
  IF STATUS2 <> 0 THEN
     WRITE "NO SUCH MEMBER ACCOUNT"  AT 15,20
     WRITE "PRESS ANY KEY TO REENTER THE MEMBER'S NUMBER"  AT 16,15
     PAUSE
     CLEAR VMEMNUM
     GOTO GETSECOND
  ENDIF

  SET VARIABLE NAME1 TO F:NAME IN #2
  SET VARIABLE NAME2 TO MID_INT IN #2
  SET VARIABLE NAME3 TO .NAME1 & .NAME2
  SET VARIABLE NAME4 TO L:NAME IN #2
  SET VARIABLE VFULNAM TO .NAME3 & .NAME4
  SET VARIABLE VSTREET TO STREET IN #2
  SET VARIABLE VCITY TO CITY IN #2
  SET VARIABLE VSTATE TO STATE IN #2
  SET VARIABLE VADDR TO .VCITY & .VSTATE
```

```
   SET VARIABLE VZIP TO ZIPCODE IN #2

   WRITE "PAYMENT SHOULD BE POSTED TO: "  AT 6,10
   WRITE .VFULNAM AT 6,40
   WRITE .VSTREET AT 7,40
   WRITE .VADDR AT 8,40
   WRITE .VZIP AT 9,40
   WRITE "IS THIS THE CORRECT MEMBER'S ACCOUNT TO POST PAYMENT TO?(Y/N)"
        AT  11,5
   FILLIN YOURANS USING " " AT 11,72
   IF YOURANS = N THEN
      CLEAR VMEMNUM
      GOTO GETSECOND
   ENDIF

   FILLIN VAMT USING "AMOUNT OF PAYMENT TO BE POSTED - "  AT 14,10
   FILLIN VCK:NO USING "PAYMENT MADE BY CHECK NUMBER - "  AT 18,10
   SET VARIABLE VINVNO TO "NONE"
   SET VARIABLE VTX:DATE TO .#DATE
   SET VARIABLE VDESCRP TO "PAYMENT CORRECTION"
   SET VHRS TO "0"
   SET VA/CNUM TO "NONE"

   LOAD MEM_CHG  *(* zero incorrect payment to account)
       .VBADMBR .VINVNO .VTX:DATE .VAMT .VDESCRP .VHRS .VA/CNUM
   END
   LOAD MEM_PAY  *(* post correct payment to account)
       .VMEMNUM .VCK:NO .VTX:DATE .VAMT PAYMENT
   END
   CLEAR VBADMBR
   CLEAR VMEMNUM
   CLEAR VAMT
   CLEAR VCK:NO
   GOTO REQAGAIN

LABEL GETBOTH
NEWPAGE
   FILLIN VBADMBR USING "PAYMENT POSTED IN ERROR TO ACCOUNT NUMBER - "
       AT 10,10
   SET POINTER #1 STATUS1 FOR MEM_REC WHERE MEM_NUM EQ .VBADMBR
   IF STATUS1 <> 0 THEN
      WRITE "NO SUCH MEMBER ACCOUNT"  AT 15,20
      WRITE "PRESS ANY KEY TO REENTER THE MEMBER'S NUMBER"  AT 16,15
      PAUSE
      CLEAR VBADMBR
      GOTO GETBOTH
   ENDIF

   SET VARIABLE NAME1 TO F:NAME IN #1
   SET VARIABLE NAME2 TO MID_INT IN #1
   SET VARIABLE NAME3 TO .NAME1 & .NAME2
   SET VARIABLE NAME4 TO L:NAME IN #1
   SET VARIABLE VFULNAM TO .NAME3 & .NAME4
   SET VARIABLE VSTREET TO STREET IN #1
   SET VARIABLE VCITY TO CITY IN #1
   SET VARIABLE VSTATE TO STATE IN #1
   SET VARIABLE VADDR TO .VCITY & .VSTATE
   SET VARIABLE VZIP TO ZIPCODE IN #1

   WRITE "PAYMENT POSTED IN ERROR TO:"  AT 15,10
   WRITE .VFULNAM AT 15,40
   WRITE .VSTREET AT 16,40
   WRITE .VADDR AT 17,40
   WRITE .VZIP AT 18,40
   WRITE "IS THIS THE ACCOUNT PAYMENT WAS POSTED IN ERROR TO?(Y/N)"
         AT 20,10
   FILLIN YOURANS USING " "  AT 20,70
   IF YOURANS = N THEN
      CLEAR VBADMBR
      GOTO GETBOTH
   ENDIF

LABEL GETNEXT
NEWPAGE
   FILLIN VMEMNUM USING "PAYMENT SHOULD HAVE BEEN MADE TO ACCOUNT - "
         AT  3,10
   SET POINTER #2 STATUS2 FOR MEM_REC WHERE MEM_NUM EQ .VMEMNUM
   IF STATUS2 <> 0 THEN
      WRITE "NO SUCH MEMBER ACCOUNT"  AT 15,20
      WRITE "PRESS ANY KEY TO REENTER THE MEMBER'S NUMBER"  AT 16,15
      PAUSE
      CLEAR VMEMNUM
      GOTO GETNEXT
   ENDIF

   SET VARIABLE NAME1 TO F:NAME IN #2
   SET VARIABLE NAME2 TO MID_INT IN #2
   SET VARIABLE NAME3 TO .NAME1 & .NAME2
   SET VARIABLE NAME4 TO L:NAME IN #2
   SET VARIABLE VFULNAM TO .NAME3 & .NAME4
   SET VARIABLE VSTREET TO STREET IN #2
   SET VARIABLE VCITY TO CITY IN #2
   SET VARIABLE VSTATE TO STATE IN #2
   SET VARIABLE VADDR TO .VCITY & .VSTATE
   SET VARIABLE VZIP TO ZIPCODE IN #2

   WRITE "PAYMENT TO BE POSTED TO: " AT 5,10
   WRITE .VFULNAM AT 5,40
   WRITE .VSTREET AT 6,40
   WRITE .VADDR AT 7,40
   WRITE .VZIP AT 8,40
   WRITE "IS THIS THE CORRECT ACCOUNT TO BE PAID?(Y/N)"  AT 10,15
   FILLIN YOURANS USING " "  AT 10,65
   IF YOUANS = N THEN
      CLEAR VMEMNUM
      GOTO GETNEXT
   ENDIF

FILLIN VBADAMT USING "PAYMENT AMOUNT ENTERED IN ERROR WAS - "  AT 12,5
   FILLIN VAMT USING "CORRECT PAYMENT AMOUNT IS - "  AT 14,5
   FILLIN VCK:NO USING "CHECK NUMBER OF PAYMENT - "  AT 16,5

   SET VARIABLE VTX:DATE TO .#DATE
   SET VARIABLE VDESCRP TO "PD COR" & .VBADMBR
   SET VARIABLE VDESCRP1 TO "PAYMENT CORRECTION"
   SET VARIABLE VDESCRP2 "PD ON ACCT" & .VMEMNUM
   SET VARIABLE VINVNO TO "NONE"
   SET VARIABLE VHRS TO "0"
   SET VARIABLE VA/CNUM TO "NONE"
   SET VARIABLE VCASH TO "1110"

   LOAD BAL_JOUR *(-- correct cash account by errored amount )
       .VCASH .VDESCRP .VBADAMT .VTX:DATE M
   END
   LOAD BAL_JOUR *(--  show correct amount received from member )
       .VCASH .VDESCRP2 .VAMT .VTX:DATE P
   END
   LOAD MEM_CHG *(-- corrective charge to zero incorrect payment entry)
       .VBADMBR .VINVNO .VTX:DATE .VBADAMT .VDESCRP1 .VHRS .VA/CNUM
   END
   LOAD MEM_PAY  *(--  show correct member's payment amount to account )
       .VMEMNUM .VCK:NO .VTX:DATE .VAMT PAYMENT
   END
   CLEAR VMEMNUM
   CLEAR VBADMBR
   CLEAR VBADAMT
   CLEAR VAMT
   CLEAR VCK:NO
```

```
  GOTO REQAGAIN

LABEL REQAGAIN
NEWPAGE
WRITE "DO YOU WISH TO EDIT ANOTHER MEMBER ACCOUNT?(Y/N)" AT 5,10
FILLIN YOURANS USING " " AT 5,60
IF YOURANS = Y THEN
   GOTO BEGINEDIT
ENDIF

LABEL EDITEND
SET MESSAGES ON
SET ERROR MESSAGES ON
RETURN
*(* END OF EDIT_PAY.CMD )

$COMMAND
EDIT_AP
*(*********************************************************************
PROGRAM:       EDIT_AP.CMD
AUTHOR:        J. M. GRAHAM
DATE:          DEC 1986      VER.  1.4
DESCRIPTION:   THIS MODULE ALLOWS EDITING OR DELETING OF PURCHASES
               MADE TO AN ACCOUNTS PAYABLE ACCOUNT.  THE MANAGER
               WILL ALSO NEED TO PROVIDE THE ACCOUNT NUMBER OF THE
               EXPENSE AND/OR ASSET ACCOUNT(S) EFFECTED BY THE CHANGE.

TABLES USED:   ACCT_PAY, BAL_JOUR, AP_CHG, EXPENSE
FORMS USED:    ADJCHGFM, ADJUSTFM, ADJEXPFM                      (08)
**********************************************************************

SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

LABEL BEGINEDIT
NEWPAGE
WRITE "PROCESSING PURCHASE CORRECTIONS TO ACCOUNTS PAYABLE ACCOUNTS" +
    AT 2,15
WRITE "-------------------------------------------------------------" +
    AT 3,15
WRITE "Corrections are handled differently if discovered prior to" +
    AT 6,10
WRITE "or after the regular end-of-month billing."  AT 7,5
WRITE "ENTER:"  AT 9,5
WRITE "1 - TO HANDLE ERRORS DISCOVERED PRIOR TO END-OF-MONTH BILLING" +
    AT 10,5
WRITE "2 - TO HANDLE ERRORS DISCOVERED AFTER END-OF-MONTH BILLING" +
    AT 12,5
WRITE "3 - EXIT WITHOUT AN UPDATE"  AT 14,5
WRITE "CHOICE: "  AT 16,5
FILLIN YOURCH USING " " AT 16,15
IF YOURCH = 1 THEN
   GOTO PROIREOM
ENDIF
IF YOURCH = 2 THEN
   GOTO AFTEREOM
ENDIF
IF YOURCH = 3 THEN
   GOTO EDITEND
ENDIF

LABEL PROIREOM
NEWPAGE
WRITE "YOU ARE ABOUT TO EDIT OR DELETE A PURCHASE TO AN ACCOUNT" +
    AT 2,10
WRITE "FOLLOW THE PROMPTS AT THE TOP OF YOUR EDIT SCREEN" AT 4,10
WRITE "TO EFFECT THE EDIT/DELETION AS YOU DESIRE." AT 5,10
WRITE "AFTER AN EDIT IS MADE YOU MUST SAVE THE EDIT BY SELECTING" +
    AT 6,10
WRITE "OR HIGHLIGHTING THE 'CHANGE' OPTION AND PRESS THE ENTER KEY" +
    AT 7,10

LABEL GETNUM
WRITE "YOU WILL NEED TO PROVIDE THE FOLLOWING DATA:" AT 9,10
WRITE "CREDITOR'S ACCOUNT NUMBER"  AT 11,20
WRITE "THE EXPENSE ACCOUNT NUMBER EFFECTED"  AT 12,20
WRITE "THE ASSET ACCOUNT NUMBER EFFECTED"  AT 13,20
WRITE "PLEASE, ENTER THE DATA AS REQUESTED"  AT 15,20
FILLIN VVENDNO USING "CREDITOR'S ACCOUNT NUMBER IS - "  AT 17,15
FILLIN VEXPACCT USING "EXPENSE ACCOUNT NUMBER EFFECTED IS - " AT 19,15
FILLIN VASSTACT USING "ASSET ACCOUNT NUMBER EFFECTED IS - "  AT 21,15

SET POINTER #1 STATUS1 FOR ACCT_PAY WHERE VENDNO EQ .VVENDNO
IF STATUS1 <> 0 THEN
   WRITE "NO SUCH CREDITOR'S ACCOUNT"  AT 22,20
   WRITE "PRESS ANY KEY TO REENTER THE CREDITOR'S NUMBER"  AT 23,15
   PAUSE
   NEWPAGE
   CLEAR VVENDNO
   GOTO GETNUM
ENDIF

   SET VARIABLE NAME1 TO VNF:NAME IN #1
   SET VARIABLE NAME2 TO VEN:MI IN #1
   SET VARIABLE NAME3 TO .NAME1 & .NAME2
   SET VARIABLE NAME4 TO VNL:NAME IN #1
   SET VARIABLE VFULNAM TO .NAME3 & .NAME4
   SET VARIABLE VSTREET TO VEN_STRT IN #1
   SET VARIABLE VCITY TO CITY IN #1
   SET VARIABLE VSTATE TO STATE IN #1
   SET VARIABLE VADDR TO .VCITY & .VSTATE
   SET VARIABLE VZIP TO ZIPCODE IN #1

   NEWPAGE
   WRITE "CREDITOR: " AT 10,20
   WRITE .VFULNAM AT 10,30
   WRITE .VSTREET  AT 11,30
   WRITE .VADDR AT 12,30
   WRITE .VZIP AT 13,30
   WRITE "IS THIS THE CORRECT ACCOUNT TO BE ADJUSTED?(Y/N)"  AT 20,20
   FILLIN YOURANS USING " "  AT 20,70
   IF YOURANS = N THEN
      NEWPAGE
      CLEAR VVENDNO
      GOTO GETNUM
   ENDIF

   EDIT USING ADJCHGFM SORTED BY VENDNO WHERE VENDNO EQ .VVENDNO

   IF VASSTACT EXISTS THEN
      EDIT USING ADJUSTFM WHERE ACCT_NUM EQ .VASSTACT
   ENDIF
   IF VEXPACCT EXISTS THEN
      EDIT USING ADJEXPFM WHERE ACCT_NUM EQ .VEXPACCT
   ENDIF

WRITE "MAKING CORRECTION TO ACCOUNT"  AT 10,20
CLEAR VVENDNO
CLEAR VASSTACT
CLEAR VEXPACCT
GOTO REQAGAIN

LABEL AFTEREOM
NEWPAGE
WRITE "ENTER:"  AT 5,10
WRITE "1 - TO CORRECT PURCHASE MADE TO WRONG CREDITOR"  AT 7,10
WRITE "2 - TO CORRECT PURCHASE AMOUNT ENTRY ERROR"  AT 9,10
WRITE "3 - BOTH THE PURCHASE AMOUNT & THE ACCOUNT CHARGED WERE WRONG" +
     AT 11,10
WRITE "4 - QUIT ... RETURN TO FINANCIAL EDIT MENU"  AT 13,10
WRITE "CHOICE:"  AT 15,10
FILLIN YOURCH USING " " AT 15,20
*(-- correction due to posting to the incorrect creditor account)
IF YOURCH = 1 THEN
   NEWPAGE
   LABEL GETFIRST
   FILLIN VBADVEN USING "PURCHASE CHARGED IN ERROR TO ACCOUNT NUMBER -" +
     AT 5,10
   SET POINTER #1 STATUS1 FOR ACCT_PAY WHERE VENDNO EQ .VBADVEN
   IF STATUS1 <> 0 THEN
      WRITE "NO SUCH CREDITOR'S ACCOUNT"  AT 15,20
      WRITE "PRESS ANY KEY TO REENTER CREDITOR'S NUMBER"  AT 16,15
      PAUSE
      CLEAR VBADVEN
      GOTO GETFIRST
   ENDIF
   SET VARIABLE NAME1 TO VNF:NAME IN #1
   SET VARIABLE NAME2 TO VEN:MI IN #1
   SET VARIABLE NAME3 TO .NAME1 & .NAME2
   SET VARIABLE NAME4 TO VNL:NAME IN #1
   SET VARIABLE VFULLNAM TO .NAME3 & .NAME4
   SET VARIABLE VSTREET TO VEN_STRT IN #1
   SET VARIABLE VCITY TO CITY IN #1
   SET VARIABLE VSTATE TO STATE IN #1
   SET VARIABLE VADDR TO .VCITY & .VSTATE
   SET VARIABLE VZIP TO ZIPCODE IN #1
   WRITE "PURCHASE CHARGED IN ERROR TO:"  AT 15,10
   WRITE .VFULLNAM AT 15,40
   WRITE .VSTREET AT 16,40
   WRITE .VADDR AT 17,40
   WRITE .VZIP AT 18,40
   WRITE "IS THIS THE ACCOUNT INCORRECTLY CHARGED?(Y/N)"  AT 20,20
   FILLIN YOURANS USING " " AT 20,70
   IF YOURANS = N THEN
      NEWPAGE
      CLEAR VBADVEN
      GOTO GETFIRST
   ENDIF
   NEWPAGE
   LABEL GETSECOND
   FILLIN VCORVEN USING "PURCHASE SHOULD HAVE BEEN CHARGED ACCOUNT -" +
      AT 3,10
   SET POINTER #2 STATUS2 FOR ACCT_PAY WHERE VENDNO EQ .VCORVEN
   IF STATUS2 <> 0 THEN
     WRITE "NO SUCH CREDITOR'S ACCOUNT"  AT 15,20
     WRITE "PRESS ANY KEY TO REENTER CREDITOR'S NUMBER TO BE PAID" +
       AT 15,10
     PAUSE
     NEWPAGE
     CLEAR VCORVEN
     GOTO GETSECOND
   ENDIF
   SET VARIABLE NAME1 TO VNF:NAME IN #2
   SET VARIABLE NAME2 TO VEN:MI IN #2
   SET VARIABLE NAME3 TO .NAME1 & .NAME2
   SET VARIABLE NAME4 TO VNL:NAME IN #2
   SET VARIABLE VFULLNAM  TO .NAME3 & .NAME4
   SET VARIABLE VSTREET TO VEN_STRT IN #2
   SET VARIABLE VCITY TO CITY IN #2
   SET VARIABLE VSTATE TO STATE IN #2
   SET VARIABLE VADDR TO .VCITY & .VSTATE
   SET VARIABLE VZIP TO ZIPCODE IN #2
   WRITE "CHARGE PURCHASE TO:"  AT 5,10
   WRITE .VFULLNAM AT 5,40
   WRITE .VSTREET AT 6,40
   WRITE .VADDR AT 7,40
   WRITE .VZIP AT 8,40
   WRITE "IS THIS THE CORRECT ACCOUNT TO BE CHARGED?(Y/N)"  AT 9,15
   FILLIN YOURANS USING " " AT 9,70
   IF YOURANS = N THEN
      NEWPAGE
      CLEAR VCORVEN
      GOTO GETSECOND
   ENDIF
   FILLIN VAMT USING "AMOUNT OF PURCHASE WAS - "  AT 12,10
   FILLIN VINVNO USING "INVOICE NUMBER OF PURCHASE WAS - " AT 14,10
   FILLIN VDATE USING "DATE OF PURCHASE WAS - "  AT 16,10
   FILLIN VPRODUCT USING "ITEM PURCHASED WAS - "  AT 18,10
   SET VARIABLE VTX:DATE TO .#DATE
   SET VARIABLE VDESCRP TO "PURCHASE CORRECTION"
   SET VARIABLE VCK:NO TO "0"
   LOAD AP_CHG  *(--show correct charge to account )
      .VCORVEN .VPRODUCT .VDATE .VAMT .VINVNO
   END
   LOAD AP_PAID  *(-- correct for incorrect charge to account )
      .VBADVEN .VTX:DATE .VAMT .VCK:NO
   END
   CLEAR VCORVEN
   CLEAR VBADVEN
   CLEAR VINVNO
   CLEAR VDATE
   CLEAR VAMT
   CLEAR VPRODUCT
   GOTO REQAGAIN
ENDIF  *(--end of correction due to incorrect charge to wrong account)

*(-- corrections due to posting incorrect purchase amount )
IF YOURCH = 2 THEN
   NEWPAGE
   LABEL GETACCT
   FILLIN VVENDNO USING "PURCHASE POSTED IN ERROR TO ACCOUNT NUMBER -" +
       AT 5,10
   SET POINTER #1 STATUS1 FOR ACCT_PAY WHERE VENDNO EQ .VVENDNO
   IF STATUS1 <> 0 THEN
     WRITE "NO SUCH CREDITOR'S ACCOUNT"  AT 15,20
     WRITE "PRESS ANY KEY TO REENTER CREDITOR'S NUMBER"  AT 16,15
     PAUSE
     NEWPAGE
     CLEAR VVENDNO
     GOTO GETACCT
   ENDIF
   SET VARIABLE NAME1 TO VNF:NAME IN #1
   SET VARIABLE NAME2 TO VEN:MI IN #1
   SET VARIABLE NAME3 TO .NAME1 & .NAME2
   SET VARIABLE NAME4 TO VNL:NAME IN #1
   SET VARIABLE VFULLNAM TO .NAME3 & .NAME4
   SET VARIABLE VSTREET TO VEN_STRT IN #1
   SET VARIABLE VCITY TO CITY IN #1
   SET VARIABLE VSTATE TO STATE IN #1
   SET VARIABLE VADDR TO .VCITY & .VSTATE
   SET VARIABLE VZIP TO ZIPCODE IN #1
   WRITE "ACCOUNT TO BE ADJUSTED IS:"  AT 7,10
   WRITE .VFULLNAM AT 7,35
   WRITE .VSTREET AT 8,35
   WRITE .VADDR AT 9,35
   WRITE .ZIP AT 10,35
   WRITE "IS THIS THE CORRECT ACCOUNT TO BE ADJUSTED?(Y/N)"  AT 11,20
   FILLIN YOURANS USING " "  AT 11,70
   IF YOURANS = N THEN
      NEWPAGE
      CLEAR VVENDNO
      GOTO GETACCT
   ENDIF
   FILLIN VBADAMT USING "PURCHASE AMOUNT ENTERED IN ERROR WAS - " +
       AT 13,10
   FILLIN VAMT USING "CORRECT PURCHASE AMOUNT TO BE POSTED IS - " +
       AT 15,10
   FILLIN VINVNO USING "INVOICE NUMBER OF PURCHASE WAS - "  AT 17,10
   FILLIN VEXPACCT USING "EXPENSE ACCOUNT NUMBER EFFECTED WAS - " +
       AT 19,10
   FILLIN VASSTACT USING "ASSET ACCOUNT NUMBER EFFECTED WAS - " +
       AT 21,10
   SET VARIABLE VTX:DATE TO .#DATE
   SET VARIABLE VDESCRP TO "PURCHASE CORRECTION"
   SET VARIABLE VCK:NO TO "0"
   LOAD AP_PAID  *(-- corrective payment to zero incorrect charge)
       .VVENDNO .VTX:DATE .VBADAMT .VCK:NO
   END
   LOAD AP_CHG *(-- show correct charge to account)
       .VVENDNO .VDESCRP .VTX:DATE .VAMT .VINVNO
   END
   IF VEXPACCT EXISTS THEN
      LOAD EXPENSE *(-- corrective minus of incorrect charge amount)
          .VEXPACCT .VDESCRP .VBADAMT .VTX:DATE M
      END
      LOAD EXPENSE  *(-- post correct increase in expense )
          .VEXPACCT .VDESCRIP .VAMT .VTX:DATE P
      END
   ENDIF
   IF VASSTACT EXISTS THEN
      LOAD BAL_JOUR  *(-- corrective minus of incorrect charge amount)
          .VASSTACT .VDESCRP .VBADAMT .VTX:DATE M
      END
      LOAD BAL_JOUR  *(-- correct increase inventory amount )
          .VASSTACT .VDESCRP .VAMT .VTX:DATE P
      END
   ENDIF
   CLEAR VVENDNO
   CLEAR VBADAMT
   CLEAR VAMT
   CLEAR VINVNO
   CLEAR EXPACCT
   CLEAR ASSTACT
   GOTO REQAGAIN
ENDIF  *(-- end of corrections due to incorrect purchase amount entry)

*( --corrections due to incorrect purchase amt and posting to wrg acct)
IF YOURCH = 3 THEN
   NEWPAGE
   LABEL ACCT1
   FILLIN VBADVEN USING"PURCHASE CHARGED IN ERROR TO ACCOUNT NUMBER -" +
       AT 5,10
   SET POINTER #1 STATUS1 FOR ACCT_PAY WHERE VENDNO EQ .VBADVEN
   IF STATUS1 <> 0 THEN
     WRITE "NO SUCH CREDITOR'S ACCOUNT"  AT 15,20
     WRITE "PRESS ANY KEY TO REENTER CREDITOR'S NUMBER"  AT 16,25
     PAUSE
     NEWPAGE
     CLEAR VBADVEN
     GOTO ACCT1
   ENDIF
   SET VARIABLE NAME1 TO VNF:NAME IN #1
   SET VARIABLE NAME2 TO VEN:MI IN #1
   SET VARIABLE NAME3 TO .NAME1 & .NAME2
   SET VARIABLE NAME4 TO VNL:NAME IN #1
   SET VARIABLE VFULLNAM TO .NAME3 & .NAME4
   SET VARIABLE VSTREET TO VEN_STRT IN #1
   SET VARIABLE VCITY TO CITY IN #1
   SET VARIABLE VSTATE TO STATE IN #1
   SET VARIABLE VADDR TO .VCITY & .VSTATE
   SET VARIABLE VZIP TO ZIPCODE IN #1
   WRITE "PURCHASE CHARGED IN ERROR TO:"  AT 15,5
   WRITE .VFULLNAM AT 15,40
   WRITE .VSTREET AT 16,40
WRITE .VADDR AT 17,40
WRITE .VZIP AT 18,40
WRITE "IS THIS THE ACCOUNT INCORRETLY CHARGED?(Y/N)"  AT 20,15
FILLIN YOURANS USING " "  AT 20,70
IF YOURANS = N THEN
   NEWPAGE
   CLEAR VBADVEN
   GOTO ACCT1
ENDIF
NEWPAGE
LABEL ACCT2
FILLIN VCORVEN USING "PURCHASE SHOULD HAVE BEEN CHARGED ACCOUNT -" +
   AT 3,10
SET POINTER #2 STATUS2 FOR ACCT_PAY WHERE VENDNO EQ .VCORVEN
IF STATUS2 <> 0 THEN
  WRITE "NO SUCH CREDITOR'S ACCOUNT"  AT 15,20
  WRITE "PRESS ANY KEY TO REENTER CREDITOR'S NUMBER TO BE PAID" +
   AT 16,10
  PAUSE
  CLEAR VCORVEN
  GOTO ACCT2
ENDIF
SET VARIABLE NAME1 TO VNF:NAME IN #2
SET VARIABLE NAME2 TO VEN:MI IN #2
SET VARIABLE NAME3 TO .NAME1 & .NAME2
SET VARIABLE NAME4 TO VNL:NAME IN #2
SET VARIABLE VFULLNAM TO .NAME3 & .NAME4
SET VARIABLE VSTREET TO VEN_STRT IN #2
SET VARIABLE VCITY TO CITY IN #2
SET VARIABLE VSTATE TO STATE IN #2
SET VARIABLE VADDR TO .VCTIY & .VSTATE
SET VARIABLE VZIP TO ZIPCODE IN #2
WRITE "CHARGE PURCHASE TO:"  AT 5,10
WRITE .VFULLNAM AT 5,40
WRITE .VSTREET AT 6,40
WRITE .VADDR AT 7,40
WRITE .ZIP AT 8,40
WRITE "IS THIS THE CORRECT ACCOUNT TO BE CHARGED?(Y/N)"  AT 9,15
FILLIN YOURANS USING " "  AT 9,70
IF YOURANS = N THEN
   NEWPAGE
   CLEAR VCORVEN
   GOTO ACCT2
ENDIF
FILLIN VBADAMT USING "PURCHASE AMOUNT ENTERED IN ERROR WAS - " +
    AT 11,10
FILLIN VAMT USING "THE CORRECT PURCHASE AMOUNT IS - " AT 13,10
FILLIN VINVNO USING "PURCHASE INVOICE NUMBER WAS - " AT 15,10
FILLIN VDATE USING "DATE OF PURCHASE WAS - " AT 17,10
FILLIN VPRODUCT USING "ITEM PURCHASED WAS - "  AT 19,10
FILLIN VEXPACCT USING "EXPENSE ACCOUNT NUMBER EFFECTED WAS - " +
    AT 21,10
FILLIN VASSTACT USING "ASSET ACCOUNT NUMBER EFFECTED WAS - " +
    AT 23,10
SET VARIABLE VCK:NO TO "0"
SET VARIABLE VTX:DATE TO .#DATE
SET VARIABLE VDESCRP TO "PURCHASE CORRECTION"
LOAD AP_PAID  *(-- corrective payment to zero incorrect charge)
    .VBADVEN .VTX:DATE .VBADAMT .VCK:NO
END
LOAD AP_CHG  *(-- show correct purchase amount and creditor )
    .VCORVEN .VPRODUCT .VDATE .VAMT .VINVNO
END
IF VEXPACCT EXISTS THEN
   LOAD EXPENSE *(--corrective minus of incorrect amount )
     .VEXPACCT .VDESCRP .VBADAMT .VTX:DATE M
   END
   LOAD EXPENSE *(-- post correct increase to expense account )
     .VEXPACCT .VDESCRP .VAMT .VTX:DATE P
```

```
         END
      ENDIF
      IF VASSTACT EXISTS THEN
         LOAD BAL_JOUR *(--corrective minus of incorrect incr to asset)
            .VASSTACT .VDESCRP .VBADAMT .VTX:DATE M
         END
         LOAD BAL_JOUR  *(-- show correct increase to inventory account)
            .VASSTACT .VDESCRP .VAMT .VTX:DATE P
         END
      ENDIF
      CLEAR VCORVEN
      CLEAR VBADVEN
      CLEAR VBADAMT
      CLEAR VAMT
      CLEAR VINVNO
      CLEAR VDATE
      CLEAR VPRODUCT
      CLEAR VEXPACCT
      CLEAR VASSTACT
      GOTO REQAGAIN
   ENDIF *(-- end of corrections for incorrect purchase and )
         *(-- wrong account charged )
   IF YOURCH = 4 THEN
      GOTO EDITEND
   ENDIF

   LABEL REQAGAIN
   NEWPAGE
   WRITE "DO YOU WISH TO EDIT ANOTHER CREDITOR'S ACCOUNT?(Y/N)" AT 5,10
   FILLIN YOURANS USING " " AT 5,65
   IF YOURANS = Y THEN
      GOTO BEGINEDIT
   ENDIF

   LABEL EDITEND
   SET MESSAGES ON
   SET ERROR MESSAGES ON
   RETURN
   *(* END OF EDIT_AP.CMD )

   $COMMAND
   EDIT_PD
   *(*********************************************************************
   PROGRAM:       EDIT_PD.CMD
   AUTHOR:        J. M. GRAHAM
   DATE:          DEC 1986      VER.  1.5
   DESCRIPTION:   THIS MODULE ALLOWS EDITING OR DELETING OF PAYMENTS
                  MADE TO AN ACCOUNTS PAYABLE ACCOUNT. MANAGER MUST
                  INDICATE WHETHER THE ERROR WAS DISCOVERED PROIR TO
                  OR AFTER THE END-OF-MONTH TRANSACTIONS HAVE BEEN
                  PROCESSED.  THE MODULE HANDLES THE TRANSACTIONS
                  DIFFERENTLY BASED UPON THE TIME OF ERROR DISCOVERY.
                  THE MANAGER MUST KNOW THE ACCOUNTS PAYABLE ACCOUNT
                  NUMBER AND PAYMENT DATA CONCERNING THE CORRECTION.

   TABLES USED:   ACCT_PAY, AP_PAID, BAL_JOUR, AP_CHG
   FORMS USED:    ADJAPDFM, ADJUSTFM                            (08)
   *********************************************************************)

   SET MESSAGES OFF
   OPEN FLYCLUB
   SET ERROR MESSAGES OFF

   LABEL BEGINEDIT
   NEWPAGE
   WRITE "PROCESSING PAYMENT CORRECTIONS TO ACCOUNTS PAYABLE ACCOUNTS" +
        AT 2,15
   WRITE "-----------------------------------------------------------" +
        AT 3,15
```

```
WRITE "Corrections are handled differently if discovered prior to" +
     AT 6,10
WRITE "or after the regular end-of-month billing."  AT 7,5
WRITE "ENTER:"  AT 9,5
WRITE "1 - TO HANDLE ERRORS DISCOVERED PRIOR TO END-OF-MONTH BILLING" +
     AT 10,5
WRITE "2 - TO HANDLE ERRORS DISCOVERED AFTER END-OF-MONTH BILLING" +
     AT 12,5
WRITE "3 - EXIT WITHOUT AN UPDATE"  AT 14,5
WRITE "CHOICE: "  AT 16,5
FILLIN YOURCH USING " " AT 16,15
IF YOURCH = 1 THEN
   GOTO PROIREOM
ENDIF
IF YOURCH = 2 THEN
   GOTO AFTEREOM
ENDIF
IF YOURCH = 3 THEN
   GOTO EDITEND
ENDIF

LABEL PROIREOM
NEWPAGE
WRITE "YOU ARE ABOUT TO EDIT OR DELETE A PAYMENT TO AN ACCOUNT" AT 2,10
WRITE "FOLLOW THE PROMPTS AT THE TOP OF YOUR EDIT SCREEN" AT 4,10
WRITE "TO EFFECT THE EDIT/DELETION AS YOU DESIRE." AT 5,10
WRITE "AFTER AN EDIT IS MADE YOU MUST SAVE THE EDIT BY SELECTING" +
     AT 6,10
WRITE "OR HIGHLIGHTING THE 'CHANGE' OPTION AND PRESS THE ENTER KEY" +
     AT 7,10

LABEL GETNUM
WRITE "YOUR MUST KNOW THE CREDITOR'S NUMBER PAYMENT WAS MADE TO" +
      AT 9,10
FILLIN VVENDNO USING "CREDITOR'S ACCOUNT NUMBER IS - "  AT 11,15
FILLIN VCK:NO USING "PAID BY CHECK NUMBER (IF KNOWN) - "  AT 13,15

SET POINTER #1 STATUS1 FOR ACCT_PAY WHERE VENDNO EQ .VVENDNO
IF STATUS1 <> 0 THEN
   WRITE "NO SUCH CREDITOR'S ACCOUNT"  AT 15,20
   WRITE "PRESS ANY KEY TO REENTER THE CREDITOR'S NUMBER"  AT 16,15
   PAUSE
   NEWPAGE
   CLEAR VVENDNO
   GOTO GETNUM
ENDIF

   SET VARIABLE NAME1 TO VNF:NAME IN #1
   SET VARIABLE NAME2 TO VEN:MI IN #1
   SET VARIABLE NAME3 TO .NAME1 & .NAME2
   SET VARIABLE NAME4 TO VNL:NAME IN #1
   SET VARIABLE VFULNAM TO .NAME3 & .NAME4
   SET VARIABLE VSTREET TO VEN_STRT IN #1
   SET VARIABLE VCITY TO CITY IN #1
   SET VARIABLE VSTATE TO STATE IN #1
   SET VARIABLE VADDR TO .VCITY & .VSTATE
   SET VARIABLE VZIP TO ZIPCODE IN #1
   SET VARIABLE VCASH TO "1110"
   SET VARIABLE VWHAT TO "PD ON ACCT" & .VVENDNO


   WRITE "CREDITOR: " AT 15,20
   WRITE .VFULNAM AT 15,30
   WRITE .VSTREET  AT 16,30
   WRITE .VADDR AT 17,30
   WRITE .VZIP AT 18,30
   WRITE "IS THIS THE CORRECT ACCOUNT TO BE ADJUSTED?(Y/N)"  AT 20,20
   FILLIN YOURANS USING " "  AT 20,70
   IF YOURANS = N THEN
```

```
      NEWPAGE
      CLEAR VVENDNO
      GOTO GETNUM
   ENDIF

IF VCK:NO EXITS THEN
   EDIT USING ADJAPDFM WHERE VENDNO EQ .VVENDNO AND CHECK:NO EQ .VCK:NO
   EDIT USING ADJUSTFM WHERE ACCT_NUM EQ .VCASH AND WHAT EQ .VWHAT
ELSE
   EDIT USING ADJAPDFM SORTED BY VENDNO WHERE VENDNO EQ .VVENDNO
   EDIT USING ADJUSTFM WHERE ACCT_NUM EQ .VCASH AND WHAT EQ .VWHAT
ENDIF
WRITE "MAKING CORRECTION TO ACCOUNT"  AT 10,20
CLEAR VVENDNO
CLEAR VCK:NO
GOTO REQAGAIN

LABEL AFTEREOM
NEWPAGE
WRITE "ENTER:"  AT 5,10
WRITE "1 - TO CORRECT PAYMENT MADE TO WRONG CREDITOR"  AT 7,10
WRITE "2 - TO CORRECT PAYMENT ENTRY ERROR"  AT 9,10
WRITE "3 - BOTH THE PAYMENT AMOUNT AND THE ACCOUNT CREDITED WERE WRONG"
      AT 11,10
WRITE "4 - QUIT ... RETURN TO FINANCIAL EDIT MENU"  AT 13,10
WRITE "CHOICE:"  AT 15,10
FILLIN YOURCH USING " " AT 15,20
*(-- correction due to posting incorrect creditor account)
IF YOURCH = 1 THEN
   NEWPAGE
   LABEL GETFIRST
   FILLIN VBADVEN USING "PAYMENT POSTED IN ERROR TO ACCOUNT NUMBER -" +
       AT 5,10
   SET POINTER #1 STATUS1 FOR ACCT_PAY WHERE VENDNO EQ .VBADVEN
   IF STATUS1 <> 0 THEN
      WRITE "NO SUCH CREDITOR'S ACCOUNT"  AT 15,20
      WRITE "PRESS ANY KEY TO REENTER CREDITOR'S NUMBER"  AT 16,15
      PAUSE
      CLEAR VBADVEN
      GOTO GETFIRST
   ENDIF
   SET VARIABLE NAME1 TO VNF:NAME IN #1
   SET VARIABLE NAME2 TO VEN:MI IN #1
   SET VARIABLE NAME3 TO .NAME1 & .NAME2
   SET VARIABLE NAME4 TO VNL:NAME IN #1
   SET VARIABLE VFULLNAM TO .NAME3 & .NAME4
   SET VARIABLE VSTREET TO VEN_STRT IN #1
   SET VARIABLE VCITY TO CITY IN #1
   SET VARIABLE VSTATE TO STATE IN #1
   SET VARIABLE VADDR TO .VCITY & .VSTATE
   SET VARIABLE VZIP TO ZIPCODE IN #1
   WRITE "PAYMENT MADE IN ERRO TO:"  AT 15,10
   WRITE .VFULLNAM AT 15,40
   WRITE .VSTREET AT 16,40
   WRITE .VADDR AT 17,40
   WRITE .VZIP AT 18,40
   WRITE "IS THIS THE ACCOUNT INCORRECTLY POSTED AS PAID?(Y/N)" +
       AT 20,15
   FILLIN YOURANS USING " " AT 20,70
   IF YOURANS = N THEN
      NEWPAGE
      CLEAR VBADVEN
      GOTO GETFIRST
   ENDIF
   NEWPAGE
   LABEL GETSECOND
   FILLIN VCORVEN USING "PAYMENT SHOULD HAVE BEEN MADE TO ACCOUNT - " +
      AT 3,10
   SET POINTER #2 STATUS2 FOR ACCT_PAY WHERE VENDNO EQ .VCORVEN
```

```
    IF STATUS2 <> 0 THEN
      WRITE "NO SUCH CREDITOR'S ACCOUNT"  AT 15,20
      WRITE "PRESS ANY KEY TO REENTER CREDITOR'S NUMBER TO BE PAID" +
       AT 15,10
      PAUSE
      NEWPAGE
      CLEAR VCORVEN
      GOTO GETSECOND
    ENDIF
    SET VARIABLE NAME1 TO VNF:NAME IN #2
    SET VARIABLE NAME2 TO VEN:MI IN #2
    SET VARIABLE NAME3 TO .NAME1 & .NAME2
    SET VARIABLE NAME4 TO VNL:NAME IN #2
    SET VARIABLE VFULLNAM  TO .NAME3 & .NAME4
    SET VARIABLE VSTREET TO VEN_STRT IN #2
    SET VARIABLE VCITY TO CITY IN #2
    SET VARIABLE VSTATE TO STATE IN #2
    SET VARIABLE VADDR TO .VCITY & .VSTATE
    SET VARIABLE VZIP TO ZIPCODE IN #2
    WRITE "PAYMENT TO BE MADE TO:"  AT 5,10
    WRITE .VFULLNAM AT 5,40
    WRITE .VSTREET AT 6,40
    WRITE .VADDR AT 7,40
    WRITE .VZIP AT 8,40
    WRITE "IS THIS THE CORRECT ACCOUNT TO BE PAID?(Y/N)"  AT 9,20
    FILLIN YOURANS USING " " AT 9,70
    IF YOURANS = N THEN
       NEWPAGE
       CLEAR VCORVEN
       GOTO GETSECOND
    ENDIF
    FILLIN VAMT USING "AMOUNT OF PAYMENT TO BE POSTED - "  AT 12,10
    FILLIN VCK:NO USING "CHECK NUMBER OF PAYMENT -" AT 14,10
    FILLIN VDATE USING "DATE OF PAYMENT WAS - "  AT 16,10
    SET VARIABLE VTX:DATE TO .#DATE
    SET VARIABLE VDESCRP TO "PAYMENT CORRECTION"
    SET VARIABLE VINVNO TO "NONE"
    *(--show payment to incorrect account; charge amt to zero acct )
    LOAD AP_CHG
       .VBADVEN .VDESCRP .VTX:DATE .VAMT .VINVNO
    END
    LOAD AP_PAID  *(-- show payment to correct account )
       .VCORVEN .VDATE .VAMT .VCK:NO
    END
    CLEAR VCORVEN
    CLEAR VBADVEN
    CLEAR VAMT
    CLEAR VCK:NO
    CLEAR VDATE
    GOTO REQAGAIN
ENDIF  *(--  end of correction due to payment to wrong account )

*(-- corrections due to posting incorrect payment amount )
IF YOURCH = 2 THEN
    NEWPAGE
    LABEL GETACCT
    FILLIN VVENDNO USING "PAYMENT AMOUNT IN ERROR IN ACCOUNT NUMBER -" +
        AT 5,10
    SET POINTER #1 STATUS1 FOR ACCT_PAY WHERE VENDNO EQ .VVENDNO
    IF STATUS1 <> 0 THEN
      WRITE "NO SUCH CREDITOR'S ACCOUNT"  AT 15,20
      WRITE "PRESS ANY KEY TO REENTER CREDITOR'S NUMBER"  AT 16,15
      PAUSE
      NEWPAGE
      CLEAR VVENDNO
      GOTO GETACCT
    ENDIF
    SET VARIABLE NAME1 TO VNF:NAME IN #1
    SET VARIABLE NAME2 TO VEN:MI IN #1
```

```
   SET VARIABLE NAME3 TO .NAME1 & .NAME2
   SET VARIABLE NAME4 TO VNL:NAME IN #1
   SET VARIABLE VFULLNAM TO .NAME3 & .NAME4
   SET VARIABLE VSTREET TO VEN_STRT IN #1
   SET VARIABLE VCITY TO CITY IN #1
   SET VARIABLE VSTATE TO STATE IN #1
   SET VARIABLE VADDR TO .VCITY & .VSTATE
   SET VARIABLE VZIP TO ZIPCODE IN #1
   WRITE "ACCOUNT TO BE ADJUSTED IS:"  AT 7,10
   WRITE .VFULLNAM AT 7,35
   WRITE .VSTREET AT 8,35
   WRITE .VADDR AT 9,35
   WRITE .ZIP AT 10,35
   WRITE "IS THIS THE CORRECT ACCOUNT TO BE ADJUSTED?(Y/N)"  AT 11,20
   FILLIN YOURANS USING " "  AT 11,70
   IF YOURANS = N THEN
      NEWPAGE
      CLEAR VVENDNO
      GOTO GETACCT
   ENDIF
   FILLIN VBADAMT USING "PAYMENT AMOUNT ENTERED IN ERROR WAS - " +
      AT 13,10
   FILLIN VAMT USING "CORRECT PAYMENT TO BE POSTED IS - "  AT 15,10
   FILLIN VCK:NO USING "PAYMENT MADE BY CHECK NUMBER - "  AT 17,10
   SET VARIABLE VTX:DATE TO .#DATE
   SET VARIABLE VDESCRP TO "PD COR" & .VVENDNO
   SET VARIABLE VDESCRP1 TO "PAYMENT CORRECTION"
   SET VARIABLE VDESCRP2 TO "PD ON ACCT" & .VVENDNO
   SET VARIABLE VCASH TO "1110"
   SET VARIABLE VINVNO TO "NONE"
   LOAD BAL_JOUR *(-- cash accounted credited for error payment amount)
       .VCASH .VDESCRP .VBADAMT .VTX:DATE P
   END
   LOAD BAL_JOUR *(-- show correct cash amount of payment)
       .VCASH .VESCRP2 .VAMT .VTX:DATE M
   END
   LOAD AP_CHG *(-- corrective charge to zero incorrect payment entry)
      .VVENDNO .VDESCRP1 .VTX:DATE .VBADAMT .VINVNO
   END
   LOAD AP_PAID  *(-- post correct payment to creditor )
      .VVENDNO .VTX:DATE .VAMT .VCK:NO
   END
   CLEAR VBADAMT
   CLEAR VAMT
   CLEAR VCK:NO
   CLEAR VVENDNO
   GOTO REQAGAIN
ENDIF  *(-- end of corrections due to incorrect payment amount entry)

   *(-- corrections due to incorrect payment amount )
   *(and posting to wrong account )
IF YOURCH = 3 THEN
   NEWPAGE
   LABEL ACCT1
   FILLIN VBADVEN USING "PAYMENT POSTED IN ERROR TO ACCOUNT NUMBER -" +
      AT 5,10
   SET POINTER #1 STATUS1 FOR ACCT_PAY WHERE VENDNO EQ .VBADVEN
   IF STATUS1 <> 0 THEN
     WRITE "NO SUCH CREDITOR'S ACCOUNT"  AT 15,20
     WRITE "PRESS ANY KEY TO REENTER CREDITOR'S NUMBER"  AT 16,25
     PAUSE
     NEWPAGE
     CLEAR VBADVEN
     GOTO ACCT1
   ENDIF
   SET VARIABLE NAME1 TO VNF:NAME IN #1
   SET VARIABLE NAME2 TO VEN:MI IN #1
   SET VARIABLE NAME3 TO .NAME1 & .NAME2
   SET VARIABLE NAME4 TO VNL:NAME IN #1
SET VARIABLE VFULLNAM TO .NAME3 & .NAME4
SET VARIABLE VSTREET TO VEN_STRT IN #1
SET VARIABLE VCITY TO CITY IN #1
SET VARIABLE VSTATE TO STATE IN #1
SET VARIABLE VADDR TO .VCITY & .VSTATE
SET VARIABLE VZIP TO ZIPCODE IN #1
WRITE "PAYMENT MADE IN ERROR TO:"  AT 15,10
WRITE .VFULLNAM AT 15,40
WRITE .VSTREET AT 16,40
WRITE .VADDR AT 17,40
WRITE .VZIP AT 18,40
WRITE "IS THIS THE ACCOUNT INCORRETLY PAID?(Y/N)"  AT 20,20
FILLIN YOURANS USING " "  AT 20,70
IF YOURANS = N THEN
   NEWPAGE
   CLEAR VBADVEN
   GOTO ACCT1
ENDIF
NEWPAGE
LABEL ACCT2
FILLIN VCORVEN USING "PAYMENT SHOULD HAVE BEEN MADE TO ACCOUNT -" +
    AT 3,10
SET POINTER #2 STATUS2 FOR ACCT_PAY WHERE VENDNO EQ .VCORVEN
IF STATUS2 <> 0 THEN
  WRITE "NO SUCH CREDITOR'S ACCOUNT"  AT 15,20
  WRITE "PRESS ANY KEY TO REENTER CREDITOR'S NUMBER TO BE PAID" +
   AT 16,10
  PAUSE
  CLEAR VCORVEN
  GOTO ACCT2
ENDIF
SET VARIABLE NAME1 TO VNF:NAME IN #2
SET VARIABLE NAME2 TO VEN:MI IN #2
SET VARIABLE NAME3 TO .NAME1 & .NAME2
SET VARIABLE NAME4 TO VNL:NAME IN #2
SET VARIABLE VFULLNAM TO .NAME3 & .NAME4
SET VARIABLE VSTREET TO VEN_STRT IN #2
SET VARIABLE VCITY TO CITY IN #2
SET VARIABLE VSTATE TO STATE IN #2
SET VARIABLE VADDR TO .VCTIY & .VSTATE
SET VARIABLE VZIP TO ZIPCODE IN #2
WRITE "PAYMENT TO BE MADE TO:"  AT 5,10
WRITE .VFULLNAM AT 5,40
WRITE .VSTREET AT 6,40
WRITE .VADDR AT 7,40
WRITE .ZIP AT 8,40
WRITE "IS THIS THE CORRECT ACCOUNT TO BE PAID?(Y/N)"  AT 9,20
FILLIN YOURANS USING " "  AT 9,70
IF YOURANS = N THEN
   NEWPAGE
   CLEAR VCORVEN
   GOTO ACCT2
ENDIF
FILLIN VBADAMT USING "PAYMENT AMOUNT ENTERED IN ERROR WAS - " +
    AT 11,10
FILLIN VAMT USING "THE CORRECT PAYMENT AMOUNT IS - " AT 13,10
FILLIN VCK:NO USING "PAYMENT MADE BY CHECK NUMBER - " AT 15,10
FILLIN VDATE USING "DATE OF PAYMENT WAS - " AT 17,10
SET VARIABLE VINVNO TO "NONE"
SET VARIABLE VTX:DATE TO .#DATE
SET VARIABLE VDESCRP TO "PAYMENT CORRECTION"
SET VARIABLE VDESCRP1 TO "PD COR" & .VBADVEN
SET VARIABLE VDESCRP2 TO "PD ON ACCT" & .VCORVEN
SET VARIABLE VCASH TO "1110"
LOAD BAL_JOUR *(-- cash account credited for error payment amount)
    .VCASH .VDESCRP1 .VBADAMT .VTX:DATE P
END
LOAD BAL_JOUR  *(-- show correct cash amount paid creditor )
    .VCASH .VDESCRP2 .VAMT .VDATE M
   END
   LOAD AP_CHG  *(--corrective charge to zero incorrect payment entry )
       .VBADVEN .VDESCRP .VTX:DATE .VBADAMT .VINVNO
   END
   LOAD AP_PAID  *(-- post correct payment to correct account )
       .VCORVEN .VDATE .VAMT .VCK:NO
   END
   CLEAR VCORVEN
   CLEAR VBADVEN
   CLEAR VBADAMT
   CLEAR VAMT
   CLEAR VCK:NO
   CLEAR VDATE
   GOTO REQAGAIN
ENDIF *(-- end of corrections due to incorrect payment )
      *(and wrong account paid )

IF YOURCH = 4 THEN
   GOTO EDITEND
ENDIF

LABEL REQAGAIN
NEWPAGE
WRITE "DO YOU WISH TO EDIT ANOTHER CREDITOR'S ACCOUNT?(Y/N)" AT 5,10
FILLIN YOURANS USING " " AT 5,70
IF YOURANS = Y THEN
   GOTO BEGINEDIT
ENDIF

LABEL EDITEND
SET MESSAGES ON
SET ERROR MESSAGES ON
RETURN
*(* END OF EDIT_PD.CMD )

$COMMAND
REV_EDIT
*(*******************************************************************
PROGRAM:      REV_EDIT.CMD
AUTHOR:       J. M. GRAHAM
DATE:         DEC 1986
DESCRIPTION:  THIS MODULE ALLOWS THE MANAGER TO MAKE COMPENSATING
              ENTRIES TO ANY REVENUE ACCOUNT.  THE MANAGER MUST KNOW
              THE REVENUE ACCOUNT NUMBER TO MAKE THE CORRECTIVE ENTRY.
              THE MODULE WILL ASK IF THE ENTRY IS A PLUS OR MINUS
              TRANSACTION AND THE AMOUNT.

TABLES USED:  EARNINGS, INCOME
FORMS USED:   ADJINCFM, ADDEARFM                              (17)
*******************************************************************)

SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

LABEL REV_MENU
NEWPAGE
WRITE "INCOME STATEMENT ACCOUNT ADJUSTMENTS" AT 5,20
WRITE "------------------------------------" AT 6,20
WRITE "ENTER:" AT 8,5
WRITE "1 - TO ENTER A NEW REVENUE ACCOUNT TO THE INCOME STATEMENT" +
     AT 9,5
WRITE "2 - TO MAKE A REVENUE ACCOUNT ADJUSTMENT" AT 10,5
WRITE "3 - TO EXIT WITHOUT AN UPDATE" AT 11,5
WRITE "YOUR CHOICE:" AT 13,5
FILLIN YRCHOICE USING " " AT 13,18
IF YRCHOICE = 1 THEN
   GOTO ADDACCT
ENDIF
```

```
IF YRCHOICE = 2 THEN
   GOTO ADJACCT
ENDIF
IF YRCHOICE = 3 THEN
   GOTO REV_END
ENDIF

*(* Module to add a new asset account to the balance sheet )
LABEL ADDACCT
NEWPAGE
ENTER ADDEARFM
GOTO REVAGAIN


*(* Draws entry form for account adjustments )
LABEL ADJACCT
ENTER ADJINCFM

*(* Loop providing for multiple entries w/o leaving module )
LABEL REVAGAIN
NEWPAGE
WRITE "DO YOU DESIRE TO MAKE ANOTHER REVENUE ACCOUNT ENTRY?" AT 8,5
WRITE "( YES OR NO ) - - (Y/N):" AT 10,20
FILLIN YOURANS USING " " AT 10,45
IF YOURANS = Y THEN
   GOTO REV_MENU
ENDIF

*(* Return control to calling module )
LABEL REV_END
SET MESSAGES ON
SET ERROR MESSAGES ON
RETURN *(-- return call to the manager's adjustment menu )
*( END OF REV_EDIT.CMD)

$COMMAND
EXP_EDIT
*(*********************************************************************
PROGRAM:      EXP_EDIT.CMD
AUTHOR:       J. M. GRAHAM
DATE:         DEC 1986
DESCRIPTION:  THIS MODULE ALLOWS THE MANAGER TO MAKE COMPENSATING
              ENTRIES TO ANY EXPENSE ACCOUNT.  THE MANAGER MUST KNOW
              THE EXPENSE ACCOUNT NUMBER TO MAKE THE CORRECTIVE ENTRY.
              THE MODULE WILL ASK IF THE ENTRY IS A PLUS OR MINUS
              TRANSACTION AND THE AMOUNT.

TABLES USED:  EARNINGS, EXPENSE
FORMS USED:   ADJEXPFM, ADDEARFM                                (17)
*********************************************************************)

SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

LABEL EXP_MENU
NEWPAGE
WRITE "INCOME STATEMENT ACCOUNT ADJUSTMENTS" AT 5,20
WRITE "------------------------------------" AT 6,20
WRITE "ENTER:" AT 8,5
WRITE "1 - TO ENTER A NEW EXPENSE ACCOUNT TO THE INCOME STATEMENT" +
     AT 9,5
WRITE "2 - TO MAKE A EXPENSE ACCOUNT ADJUSTMENT" AT 10,5
WRITE "3 - TO EXIT WITHOUT AN UPDATE" AT 11,5
WRITE "YOUR CHOICE:" AT 13,5
FILLIN YRCHOICE USING " " AT 13,18
IF YRCHOICE = 1 THEN
   GOTO ADDACCT
ENDIF
```

```
IF YRCHOICE = 2 THEN
   GOTO ADJACCT
ENDIF
IF YRCHOICE = 3 THEN
   GOTO EXP_END
ENDIF

*(* Module to add a new asset account to the balance sheet )
LABEL ADDACCT
NEWPAGE
ENTER ADDEARFM
GOTO EXPAGAIN


*(* Draws entry form for account adjustments )
LABEL ADJACCT
ENTER ADJEXPFM

*(* Loop providing for multiple entries w/o leaving module )
LABEL EXPAGAIN
NEWPAGE
WRITE "DO YOU DESIRE TO MAKE ANOTHER EXPENSE ACCOUNT ENTRY?" AT 8,5
WRITE "( YES OR NO ) - - (Y/N):" AT 10,20
FILLIN YOURANS USING " " AT 10,45
IF YOURANS = Y THEN
   GOTO EXP_MENU
ENDIF

*(* Return control to calling module )
LABEL EXP_END
SET MESSAGES ON
SET ERROR MESSAGES ON
RETURN *(-- return call to the manager's adjustment menu )
*( END OF EXP_EDIT.CMD)
$COMMAND
ASSTEDIT
*(**********************************************************************
PROGRAM:       ASSTEDIT.CMD
AUTHOR:        J. M. GRAHAM
DATE:          DEC 1986
DESCRIPTION:   THIS MODULE ALLOWS THE MANAGER TO MAKE COMPENSATING
               ENTRIES TO ANY ASSET ACCOUNT.  THE MANAGER MUST KNOW
               THE ASSET ACCOUNT NUMBER TO MAKE THE CORRECTIVE ENTRY.
               THE MODULE WILL ASK IF THE ENTRY IS A PLUS OR MINUS
               TRANSACTION AND THE AMOUNT.

TABLES USED:   BAL_SHET, BAL_JOUR
FORMS USED:    ADJUSTFM, ADDBALFM                               (17)
**********************************************************************)

SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

LABEL ASSTMENU
NEWPAGE
WRITE "BALANCE SHEET ASSET ACCOUNT ADJUSTMENTS" AT 5,20
WRITE "---------------------------------------" AT 6,20
WRITE "ENTER:" AT 8,5
WRITE "1 - TO ENTER A NEW ASSET ACCOUNT TO THE BALANCE SHEET" AT 9,5
WRITE "2 - TO MAKE AN ASSET ACCOUNT ADJUSTMENT" AT 10,5
WRITE "3 - TO EXIT WITHOUT AN UPDATE" AT 11,5
WRITE "YOUR CHOICE:" AT 13,5
FILLIN YRCHOICE USING " " AT 13,18
IF YRCHOICE = 1 THEN
   GOTO ADDACCT
ENDIF
IF YRCHOICE = 2 THEN
   GOTO ADJACCT
```

```
ENDIF
IF YRCHOICE = 3 THEN
   GOTO ASSTEND
ENDIF

*(* Module to add a new asset account to the balance sheet )
LABEL ADDACCT
NEWPAGE
ENTER ADDBALFM
GOTO ASTAGAIN


*(* Draws entry form for account adjustments )
LABEL ADJACCT
ENTER ADJUSTFM

*(* Loop providing for multiple entries w/o leaving module )
LABEL ASTAGAIN
NEWPAGE
WRITE "DO YOU DESIRE TO MAKE ANOTHER ASSET ACCOUNT ENTRY?" AT 8,5
WRITE "( YES OR NO ) - - (Y/N):" AT 10,20
FILLIN YOURANS USING " " AT 10,45
IF YOURANS = Y THEN
   GOTO ASSTMENU
ELSE
   GOTO ASSTEND
ENDIF

*(* Return control to calling module )
LABEL ASSTEND
SET MESSAGES ON
SET ERROR MESSAGES ON
RETURN *(-- return call to the manager's adjustment menu )
*( END OF ASSTEDIT.CMD)
$COMMAND
LIABEDIT
*(*********************************************************************
PROGRAM:      LIABEDIT.CMD
AUTHOR:       J. M. GRAHAM
DATE:         DEC 1986
DESCRIPTION:  THIS MODULE ALLOWS THE MANAGER TO MAKE COMPENSATING
              ENTRIES TO ANY LIABILITY ACCOUNT.  THE MANAGER MUST KNOW
              THE ACCOUNT NUMBER TO MAKE THE CORRECTIVE ENTRY.
              THE MODULE WILL ASK IF THE ENTRY IS A PLUS OR MINUS
              TRANSACTION AND THE AMOUNT.

TABLES USED:  BAL_SHET, BAL_JOUR
FORMS USED:   ADJUSTFM, ADDBALFM                                  (17)
*********************************************************************)

SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

LABEL LIABMENU
NEWPAGE
WRITE "BALANCE SHEET LIABILITY ACCOUNT ADJUSTMENTS" AT 5,15
WRITE "-------------------------------------------" AT 6,15
WRITE "ENTER:" AT 8,5
WRITE "1 - TO ENTER A NEW LIABILITY ACCOUNT TO THE BALANCE SHEET" +
      AT 9,5
WRITE "2 - TO MAKE AN LIABILITY ACCOUNT ADJUSTMENT" AT 10,5
WRITE "3 - TO EXIT WITHOUT AN UPDATE" AT 11,5
WRITE "YOUR CHOICE:" AT 13,5
FILLIN YRCHOICE USING " " AT 13,18
IF YRCHOICE = 1 THEN
   GOTO ADDACCT
ENDIF
IF YRCHOICE = 2 THEN
```

```
   GOTO ADJACCT
ENDIF
IF YRCHOICE = 3 THEN
   GOTO LIABEND
ENDIF

*(* Module to add a new asset account to the balance sheet )
LABEL ADDACCT
NEWPAGE
ENTER ADDBALFM
GOTO LIBAGAIN


*(* Draws entry form for account adjustments )
LABEL ADJACCT
ENTER ADJUSTFM

*(* Loop providing for multiple entries w/o leaving module )
LABEL LIBAGAIN
NEWPAGE
WRITE "DO YOU DESIRE TO MAKE ANOTHER ASSET ACCOUNT ENTRY?" AT 8,5
WRITE "( YES OR NO ) - - (Y/N):" AT 10,20
FILLIN YOURANS USING " " AT 10,45
IF YOURANS = Y THEN
   GOTO LIABMENU
ELSE
   GOTO LIABEND
ENDIF

*(* Return control to calling module )
LABEL LIABEND
SET MESSAGES ON
SET ERROR MESSAGES ON
RETURN *(-- return call to the manager's adjustment menu )
*( END OF LIABEDIT.CMD)
```

APPENDIX G
CLUB_PRT SOURCE CODE

```
*(*******************************************************************)
*(*                   MONTEREY NAVY FLYING CLUB                     *)
*(*                MANAGEMENT INFORMATION SYSTEM                    *)
*(*                   PROTOTYPE VERSION 2.0                         *)
*(*                                                                 *)
*(*                   PRINTING APPLICATION                          *)
*(*                                                                 *)
*(*                 LCDR. JAMES M. GRAHAM, USN                      *)
*(*                        JANUARY 1987                        (07) *)
*(*******************************************************************)

$COMMAND
CLUB_PRT
SET MESSAGE OFF
OPEN FLYCLUB
SET ERROR MESSAGE OFF
SET VAR PICK1  INT
LABEL STARTAPP
  NEWPAGE
  CHOOSE PICK1  FROM MAIN     IN CLUB_PRT.APX
  IF PICK1  EQ 0 THEN
    GOTO ENDAPP
  ENDIF
  IF PICK1  EQ            1 THEN
    SET VAR PICK2  INT
    SET VAR LEVEL2 INT
    SET VAR LEVEL2 TO 0
    WHILE LEVEL2 EQ 0  THEN
      NEWPAGE
      CHOOSE PICK2  FROM ADMINPRT IN CLUB_PRT.APX
      IF PICK2  EQ 0 THEN
        BREAK
      ENDIF
      IF PICK2  EQ            1 THEN
        RUN ROSTER   IN CLUB_PRT.APX
      ENDIF
      IF PICK2  EQ            2 THEN
        RUN MGR_PRT  IN CLUB_PRT.APX
      ENDIF
      IF PICK2  EQ            3 THEN
        RUN ACSTAT   IN CLUB_PRT.APX
      ENDIF
      IF PICK2  EQ            4 THEN
        BREAK
      ENDIF
    ENDWHILE
    CLEAR LEVEL2
    CLEAR PICK2
    GOTO STARTAPP
  ENDIF
  IF PICK1  EQ            2 THEN
    SET VAR PICK2  INT
    SET VAR LEVEL2 INT
    SET VAR LEVEL2 TO 0
    WHILE LEVEL2 EQ 0  THEN
      NEWPAGE
      CHOOSE PICK2  FROM FINPRT   IN CLUB_PRT.APX
      IF PICK2  EQ 0 THEN
        BREAK
      ENDIF
      IF PICK2  EQ            1 THEN
```

```
        RUN AGED_AR  IN CLUB_PRT.APX
      ENDIF
      IF PICK2  EQ             2 THEN
        RUN PRT_BAL  IN CLUB_PRT.APX
      ENDIF
      IF PICK2  EQ             3 THEN
        RUN PRT_INC  IN CLUB_PRT.APX
      ENDIF
      IF PICK2  EQ             4 THEN
        BREAK
      ENDIF
    ENDWHILE
    CLEAR LEVEL2
    CLEAR PICK2
    GOTO STARTAPP
  ENDIF
  IF PICK1  EQ             3 THEN
    SET VAR PICK2  INT
    SET VAR LEVEL2 INT
    SET VAR LEVEL2 TO 0
    WHILE LEVEL2 EQ 0  THEN
      NEWPAGE
      CHOOSE PICK2  FROM YRLYRPT  IN CLUB_PRT.APX
      IF PICK2  EQ 0 THEN
        BREAK
      ENDIF
      IF PICK2  EQ             1 THEN
        RUN PRTMEMBD IN CLUB_PRT.APX
      ENDIF
      IF PICK2  EQ             2 THEN
        RUN OP_STMT  IN CLUB_PRT.APX
      ENDIF
      IF PICK2  EQ             3 THEN
        RUN INS_SUM  IN CLUB_PRT.APX
      ENDIF
      IF PICK2  EQ             4 THEN
        BREAK
      ENDIF
    ENDWHILE
    CLEAR LEVEL2
    CLEAR PICK2
    GOTO STARTAPP
  ENDIF
  IF PICK1  EQ             4 THEN
    GOTO ENDAPP
  ENDIF
  GOTO STARTAPP
LABEL ENDAPP
CLEAR PICK1
RETURN
$MENU
MAIN
COLUMN PRINT REPORTS / STATEMENTS
PRINT ADMINISTATIVE REPORTS
PRINT FINANCIAL STATEMENTS
PRINT ANNUAL REPORT INPUTS
RETURN TO THE MNFC MAIN MENU
$MENU
ADMINPRT
COLUMN PRINT ADMINISTRATIVE REPORTS
PRINT MEMBER ROSTER
PRINT MANAGER'S MONTHLY REPORT
PRINT A/C INVENTORY & STATUS REPORT
RETURN TO PRINT REPORTS / STATEMENTS MENU
$MENU
YRLYRPT
COLUMN PRINT ANNUAL REPORT INPUTS
PRINT MEMBERSHIP BREAKDOWN
PRINT ANNUAL OPERATING STATEMENT INPUTS
PRINT SUMMARY OF INSURANCE
RETURN TO PRINT REPORTS / STATEMENTS MENU
$MENU
FINPRT
COLUMN PRINT FINANCIAL STATEMENTS
PRINT AGED ACCOUNTS RECEIVABLE
PRINT BALANCE SHEET
PRINT INCOME STATEMENT
RETURN TO PRINT REPORTS / STATEMENTS MENU
$COMMAND
ROSTER
*(*********************************************************************
PROGRAM:       ROSTER.CMD
AUTHOR:        J. M. GRAHAM
DATE:          JAN 1987  VER. 1.4
DESCRIPTION:   PRINTS THE MONTEREY NAVY FLYING CLUB MEMBERSHIP ROSTER
               INCLUDING BOTH ACTIVE AND INACTIVE MEMBERSHIP.

TABLE USED:    MEM_REC
FORMS USED:    NONE
REPORT USED:   CLUB ROSTER                                  (19)
********************************************************************)

SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

NEWPAGE
WRITE "THE CLUB ROSTER MAY BE PRINTED AS OFTEN AS DESIRED." AT 5,15
WRITE "UPDATES ARE PERFORMED WHENEVER MEMBERS ARE ENTERED." AT 7,15
WRITE "DO YOU WANT TO PROCEED? (Y/N) - " AT 11,26
FILLIN YOURANS USING " " AT 11,56
IF YOURANS = N THEN
   GOTO BAILOUT
ENDIF

NEWPAGE
WRITE "ENSURE PRINTER IS READY." AT 10,28
WRITE "PRESS ANY KEY TO BEGIN PRINTING OF ROSTER."  AT 15,19
PAUSE

NEWPAGE
WRITE "WORKING ON PRINTING THE MEMBERSHIP ROSTER"  AT 10,19
WRITE "THIS WILL TAKE A LITTLE TIME, WHY NOT TAKE FIVE ON ME." AT 15,13

*(-- Print club roster )
OUTPUT PRINTER
SET VARIABLE TOTNUM INTEGER
COMPUTE TOTNUM AS COUNT MEM_NUM FROM MEM_REC
PRINT ROSTER SORTED BY L:NAME F:NAME
OUTPUT SCREEN

NEWPAGE
WRITE "CLUB ROSTER COMPLETE." AT 7,29
WRITE "PRESS ANY KEY TO RETURN TO MENU." AT 10,24
PAUSE

*(-- Bailout routine for immediate exit)
LABEL BAILOUT
  SET MESSAGES ON
  SET ERROR MESSAGES ON
  RETURN
*(* END OF ROSTER.CMD )

$COMMAND
MGR_PRT
*(*********************************************************************
PROGRAM:       MGR_PRT.CMD
AUTHOR:        J. M. GRAHAM
DATE:          JAN 1987     VER 1.3
DESCRIPTION:   THIS MODULE PREPARES THE MONTHLY MANAGER'S REPORT.
               THE REPORT MUST BE COMPLETED AFTER THE MONTHLY BILLING.

TABLES USED:   BAL_SHET, EARNINGS, FLT_HIST, MEM_REC
FORMS USED:    NONE
REPORTS USED:  MGR_RPT                                           (19)
*******************************************************************)

SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

NEWPAGE
WRITE "THE MANAGER'S REPORT MUST BE PRINTED AFTER THE MONTHLY" AT 5,12
WRITE "BILLING AND END-OF-THE-MONTH POSTINGS HAVE BEEN DONE." AT 6,13
WRITE "DO YOU WANT TO PROCEED? (Y/N) - " AT 10,24
FILLIN YOURANS USING " " AT 10,58
IF YOURANS = N THEN
   GOTO MGREND
ENDIF

NEWPAGE
SET VARIABLE VMONTH TEXT
SET VARIABLE VSTART DATE
SET VARIABLE VEND DATE
WRITE "THE MANAGER'S REPORT IS TO COVER WHICH MONTH?"  AT 5,17
FILLIN VMONTH USING "MONTH:  "  AT 7,35
WRITE "IN ORDER TO COMPUTE THE MONTHLY FLIGHT HOURS"  AT 13,25
WRITE "PLEASE, ENTER THE START DATE AND END DATE OF THE MONTH"  AT 14,20
WRITE "THAT THIS MANAGER'S REPORT COVERS"  AT 15, 23
WRITE "EXAMPLE:  START:  1/1/87"  AT 17,25
WRITE "END:  1/31/87"  AT 18,25

FILLIN VSTART USING "THE DATE OF MONTH'S START:  "  AT 20, 25
FILLIN VEND USING "THE DATE OF MONTH'S END:  "  AT 22,25
NEWPAGE
WRITE "ENSURE PRINTER IS READY." AT 10,28
WRITE "PRESS ANY KEY TO BEGIN PRINTING THE MANAGER'S REPORT."  AT 15,14
PAUSE

NEWPAGE
WRITE "WORKING ON PRINTING THE MANAGER'S REPORT"  AT 10,20
WRITE "THIS WILL TAKE A LITTLE TIME, WHY NOT TAKE FIVE ON ME." AT 15,13

*(-- Print MANAGER'S REPORT )
*(-- compute the total membership )
COMPUTE VTOTMBRS AS COUNT MEM_NUM FROM MEM_REC
*(-- compute the month's flt hours as per the month dates given)
COMPUTE VFLTHRS AS SUM FLT_HRS FROM FLT_HIST WHERE FLTDATE >= .VSTART +
    AND FLTDATE <= .VEND
SET VARIABLE VMONFLT TO .VFLTHRS
*(--  separate income from earnings table to compute income)
PROJECT TEMP1 FROM EARNINGS USING ACCT_NUM BALANCE WHERE +
     ACCT_NUM >= "4941" AND ACCT_NUM <= "4952"
APPEND EARNINGS TO TEMP1 WHERE ACCT_NUM >= "8100" AND ACCT_NUM +
     <= "8301"
COMPUTE VMONINC AS SUM BALANCE FROM TEMP1
REMOVE TEMP1
*(-- separate expenses from earnings table to compute expense only)
PROJECT TEMP2 FROM EARNINGS USING ACCT_NUM BALANCE WHERE +
     ACCT_NUM >= "5941" AND ACCT_NUM <= 7930"
APPEND EARNINGS TO TEMP2 WHERE ACCT_NUM = "9255"
COMPUTE VMONEXP AS SUM BALANCE FROM TEMP2
REMOVE TEMP2
*(-- withdraw from balance sheet tabel the asset and liab data)
COMPUTE VCHECK AS SUM BALANCE FROM BAL_SHET WHERE ACCT_NUM = "1110"
COMPUTE VSAV AS SUM BALANCE FROM BAL_SHET WHERE ACCT_NUM = "1130"
COMPUTE VACMBRS AS SUM BALANCE FROM BAL_SHET WHERE ACCT_NUM = "1210"
COMPUTE VACLESR AS SUM BALANCE FROM BAL_SHET WHERE ACCT_NUM = "1211"
COMPUTE VAP AS SUM BALANCE FROM BAL_SHET WHERE ACCT_NUM = "2100"
COMPUTE VSHTERM AS SUM BALANCE FROM BAL_SHET WHERE ACCT_NUM = "2510"

OUTPUT PRINTER
PRINT MGR_RPT
OUTPUT SCREEN

CLEAR VMONFLT
CLEAR VMONINC
CLEAR VMONEXP
CLEAR VTOTMBRS
CLEAR VCHECK
CLEAR VSAV
CLEAR VACMBRS
CLEAR VACLESR
CLEAR VTOTMBRS
CLEAR VAP
CLEAR VSHTERM


NEWPAGE
WRITE "MANAGER'S REPORT COMPLETE." AT 7,27
WRITE "PRESS ANY KEY TO RETURN TO MENU." AT 10,24
PAUSE

LABEL MGREND
SET MESSAGES ON
SET ERROR MESSAGES ON
RETURN
*(* END OF MGR_RPT.CMD )



$COMMAND
ACSTAT
*(*********************************************************************
PROGRAM:      ACSTAT.CMD
AUTHOR:       J. M. GRAHAM
DATE:         JAN 1987
DESCRIPTION:  THIS MODULE PRINTS THE AIRCRAFT INVENTORY AND STATUS
              REPORT.  UPDATES THE A/C_HRS TABLE AND READYS IT FOR
              THE NEXT MONTH'S ENTRIES.

TABLES USED:  A/C_HRS
FORMS USED:   FLTHRSFM
REPORTS USED: A/C_STAT                                         (11)
*********************************************************************)

SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

NEWPAGE
WRITE "AIRCRAFT STATUS REPORTS SHOULD BE PRINTED AT MONTH'S END." AT 5,8
WRITE "AIRCRAFT HOURS ARE UPDATED PER A/C TRANSACTION." AT 7,12
WRITE "ENSURE A BACKUP COPY OF THE DATABASE HAS BEEN MADE." AT 9,10
WRITE "DO YOU WANT TO PROCEED? (Y/N) - " AT 11,17
FILLIN YOURANS USING " " AT 11,50
IF YOURANS = N THEN
   GOTO STATEND
ENDIF

*(-- allowing manager to verify a/c hours and update a/c_hrs )
FILLIN VLOOK USING "DO YOU WANT TO VERIFY THE A/C HOURS?(Y/N)" AT 16,10
IF VLOOK = Y THEN
   EDIT USING FLTHRSFM SORTED BY A/C_NUM
ENDIF
```

```
NEWPAGE
WRITE "ENSURE PRINTER IS READY, THEN PRESS ANY KEY TO CONTINUE." AT 10,5
PAUSE

NEWPAGE
WRITE "PRINTING THE AIRCRAFT INVENTORY AND STATUS REPORT"  AT 10,10
WRITE "THIS WILL TAKE A LITTLE TIME...WHY NOT TAKE FIVE"  AT 15,10

*(-- print a/c status report )
OUTPUT PRINTER
SET NULL " "
PRINT A/C_STAT
OUTPUT SCREEN
SET NULL -0-

*(-- Establish temporary table to maintain copy of CUM_HRS, HOB_END, and
     MEM_NUM when tables are cleared for next month's entries.)
DELETE ROWS FROM TEMP_HRS WHERE LIMIT = 100
APPEND A/C_HRS TO TEMP_HRS
*(-- Move transactions to history tables)
APPEND A/C_HRS TO HOURHIST

*(-- Prepare tables for next month's entries)
DELETE ROWS FROM A/C_HRS WHERE A/C_NUM EXISTS

*(-- Return CUM_HRS, HOB_END, and MEM_NUM to A/C_HRS )
APPEND TEMP_HRS TO A/C_HRS
*(-- Updates complete)

NEWPAGE
WRITE "Monthly A/C Inventory and Status Report is Complete." AT 7,5
WRITE "PRESS ANY KEY TO CONTINUE" AT 10,18
PAUSE

LABEL STATEND
SET MESSAGES ON
SET ERROR MESSAGES ON
RETURN
*(*  END OF ACSTAT.CMD )

$COMMAND
PRTMEMBD
*(**********************************************************************
PROGRAM:       PRTMEMBD.CMD
AUTHOR:        J. M. GRAHAM
DATE:          JAN 1987      VER 1.3
DESCRIPTION:   THIS MODULE PREPARES THE MEMBERSHIP BREAKDOWN REPORT
               SHOWING WHICH CATEGORY OF MEMBER, SUCH AS NAVAL OFFICER
               ARMY OFFICER, HAS FLOWN HOW MANY HOURS OVER THE PAST
               YEAR.

TABLES USED:   MBR_SUM
FORMS USED:    NONE
REPORT USED:   MBR_BKDN                                          (16)
*********************************************************************)

SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

NEWPAGE
WRITE "MEMBERSHIP FLIGHT BREAKDOWN MAY BE PRINTED ANY TIME." AT 5,5
WRITE "HOWEVER, IT WILL BE ACCURATE AS OF THE LAST MONTH POSTING"AT 7,5
WRITE "AND SHOULD BE PRINTED AFTER THE END OF THE MONTH."  AT 8,5
WRITE "DO YOU WANT TO PROCEED? (Y/N) - " AT 15,17
FILLIN YOURANS USING " " AT 15,50
IF YOURANS = N THEN
   GOTO PRTMBREND
ENDIF
```

```
NEWPAGE
WRITE "!!!! WARNING !!!!"  AT 5,20
WRITE "AFTER PRINTING THE YEARLY REPORT THE DATA IS DELETED"  AT 8,10
WRITE "YOU CAN PRINT A MONTHLY REPORT OR A YEARLY REPORT"  AT 10,10
WRITE "ENTER:"  AT 12,20
WRITE "1 - MONTHLY REPORT"  AT 14,20
WRITE "2 - YEARLY REPORT"  AT 16,20
WRITE "3 - EXIT WITHOUT A REPORT "  AT 18,20
FILLIN YOURCH USING "YOUR CHOICE IS - "  AT 20,20
IF YOURCH = 1 THEN
   GOTO MONTHLY
ENDIF
IF YOURCH = 2 THEN
   GOTO YEARLY
ENDIF
IF YOURCH = 3 THEN
   GOTO PRTMBREND
ENDIF
LABEL MONTHLY
NEWPAGE
WRITE "ENSURE PRINTER IS READY, THEN PRESS ANY KEY TO CONTINUE." AT 10,5
PAUSE

NEWPAGE
WRITE "PRINTING THE MEMBERSHIP FLIGHT BREAKDOWN REPORT"  AT 10,15
WRITE "THIS WILL TAKE A LITTLE TIME, WHY NOT TAKE FIVE ON ME." AT 15,8

*(-- Print membership flight breakdown report)
OUTPUT PRINTER
*(-- provides manager with hours for his information )
SELECT CATEGORY CATNAME=16 STUDCT=3 STUDHRS PRIVCT=3 PRIVHRS COMMCT=3 +
  COMMHRS CFICT=3 CFIHRS TOTCOUNT=4 TOTHOURS FROM MBR_SUM
*(-- report does not have hours of each cert; just number in club)
NEWPAGE
PRINT MBR_BKDN
OUTPUT SCREEN

NEWPAGE
WRITE "MEMBERSHIP FLIGHT BREAKDOWN REPORT COMPLETE." AT 7,10
WRITE "PRESS ANY KEY TO RETURN TO MENU." AT 10,15
PAUSE
GOTO PRTMBREND


LABEL YEARLY
NEWPAGE
WRITE "ENSURE PRINTER IS READY, THEN PRESS ANY KEY TO CONTINUE" AT 10,5
PAUSE

OUTPUT PRINTER
*(-- prints for manager hours and numbers for reference later)
SELECT CATEGORY CATNAME=16 STUDCT=3 STUDHRS PRIVCT=3 PRIVHRS COMMCT=3 +
  COMMHRS CFICT=3 CFIHRS TOTCOUNT=4 TOTHOURS FROM MBR_SUM
*(-- print of report )
NEWPAGE
PRINT MBR_BKDN
OUTPUT SCREEN

WRITE "PLEASE, STAY BY WHILE I ZERO THE MEMBERSHIP SUMMARY"  AT 15,10
WRITE "TABLE AND PREPARE IT FOR THE NEW YEAR'S DATA."  AT 17,13

*(-- delete data from the MBR_SUM TABLE preparing it for the next year)
SET VARIABLE VENDCT INTEGER
SET VARIABLE VENDHRS REAL
SET VARIABLE VENDCT TO 0
SET VARIABLE VENDHRS TO 0

SET POINTER #1 STATUS1 FOR MBR_SUM
```

```
  WHILE STATUS1 = 0 THEN
    SET VARIABLE TEMPCAT TO CATEGORY IN #1
    CHANGE STUDCT TO .VENDCT IN MBR_SUM WHERE CATEGORY = .TEMPCAT
    CHANGE STUDHRS TO .VENDHRS IN MBR_SUM WHERE CATEGORY = .TEMPCAT
    CHANGE PRIVCT TO .VENDCT IN MBR_SUM WHERE CATEGORY = .TEMPCAT
    CHANGE PRIVHRS TO .VENDHRS IN MBR_SUM WHERE CATEGORY = .TEMPCAT
    CHANGE COMMCT TO .VENDCT IN MBR_SUM WHERE CATEGORY = .TEMPCAT
    CHANGE COMMHRS TO .VENDHRS IN MBR_SUM WHERE CATEGORY = .TEMPCAT
    CHANGE CFICT TO .VENDCT IN MBR_SUM WHERE CATEGORY = .TEMPCAT
    CHANGE CFIHRS TO .VENDHRS IN MBR_SUM WHERE CATEGORY = .TEMPCAT
    CHANGE TOTCOUNT TO .VENDCT IN MBR_SUM WHERE CATEGORY = .TEMPCAT
    CHANGE TOTHOURS TO .VENDHRS IN MBR_SUM WHERE CATEGORY = .TEMPCAT
    NEXT #1 STATUS1
  ENDWHILE *(-- end of preparing table for next month)

LABEL PRTMBREND
SET MESSAGES ON
SET ERROR MESSAGES ON
RETURN
*(* END OF PRTMEMBD.CMD )

$COMMAND
OP_STMT
*(*******************************************************************
PROGRAM:      OP_STMT.CMD
AUTHOR:       J. M. GRAHAM
DATE:         JAN 1987     VER 1.5
DESCRIPTION:  THIS MODULE PRODUCES ROUGH INPUTS FOR THE REQUIRED
              OPERATING STATEMENTS INCLUDING STATEMENT OF NET WORTH
              FOR THE ANNUAL REPORT INPUT IN ACCCORDANCE WITH CNO
              INST.  THIS REPORT MUST BE PRINTED AFTER THE MONTHLY
              POSTINGS INORDER FOR THE DATA TO BE ACCURATE.

              THE REPORT IS LISTED AS FROM TEMP_HRS TABLE, BUT THIS
              IS ONLY TO ALLOW THE USE OF THE REPORT FORMATER TO USE
              DATA COMPUTED IN THIS MODULE IN THE REPORT.

TABLES USED:  BAL_SHET, EARNINGS
FORMS USED:   NONE
REPORT USED:  OPSRPT1, OPSRPT2, OPSRPT3, OPSRPT4, OPSRPT5  (19)
*******************************************************************)

SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

NEWPAGE
WRITE "OPERATING STATEMENTS SHOULD BE PRINTED AFTER THE MONTH" AT 5,16
WRITE "OF SEPTEMBER'S BILLING AND POSTING TO END THE FY."  AT 7,17
WRITE "THEY ARE DESIGNED AS INPUTS TO REVIEW AND ASSIST IN" AT 10,16
WRITE "COMPLETING YOUR ANNUAL OFFICAL REPORTS."  AT 12,21
WRITE "DO YOU WANT TO PROCEED? (Y/N) - " AT 16,17
FILLIN YOURANS USING " " AT 16,50
IF YOURANS = N THEN
   GOTO OPSMTEND
ENDIF

NEWPAGE
SET VARIABLE VYEAR TEXT
FILLIN VYEAR USING "REPORTS ARE FOR YEAR 19 -"  AT 5,25
WRITE "ENSURE PRINTER IS READY." AT 10,24
WRITE "PRESS ANY KEY TO BEGIN PRINTING THE REPORT INPUTS"  AT 15,15
PAUSE

NEWPAGE
WRITE "PRINTING THE OPERATING STATEMENTS"  AT 10,20
WRITE "THIS WILL TAKE A LITTLE TIME, WHY NOT TAKE FIVE ON ME." AT 15,8

*(-- Print operating statements )
```

```
OUTPUT PRINTER
*(-- printing the first of the balance sheets)
COMPUTE VCASH AS SUM BALANCE FROM BAL_SHET WHERE ACCT_NUM = "1110"
COMPUTE VSAV AS SUM BALANCE FROM BAL_SHET WHERE ACCT_NUM = "1130"
COMPUTE VAR AS SUM BALANCE FROM BAL_SHET WHERE ACCT_NUM = "1210" +
   AND ACCT_NUM = "1211"
COMPUTE VINVENT AS SUM BALANCE FROM BAL_SHET WHERE ACCT_NUM = "1341"
COMPUTE VEQUIP AS SUM BALANCE FROM BAL_SHET WHERE ACCT_NUM = "1651"
COMPUTE VEQDEP AS SUM BALANCE FROM BAL_SHET WHERE ACCT_NUM = "1751"
COMPUTE VA/C AS SUM BALANCE FROM BAL_SHET WHERE ACCT_NUM = "1650"
COMPUTE VA/CDEP AS SUM BALANCE FROM BAL_SHET WHERE ACCT_NUM = "1750"
COMPUTE VPREPD AS SUM BALANCE FROM BAL_SHET WHERE ACCT_NUM = "1530"
COMPUTE VTOTASET AS SUM BALANCE FROM BAL_SHET WHERE ACCT_NUM >= "1110" +
   AND ACCT_NUM <= "1751"
PRINT OPSRPT1
NEWPAGE       *(-- makes print form feed paper after printing)
CLEAR VCASH
CLEAR VSAV
CLEAR VAR
CLEAR VINVENT
CLEAR VEQUIP
CLEAR VEQDEP
CLEAR VA/C
CLEAR VA/CDEP
CLEAR VPREPD

*(-- printing the last of the balance sheets)
COMPUTE VAP AS SUM BALANCE FROM BAL_SHET WHERE ACCT_NUM = "2100"
COMPUTE VMISCL AS SUM BALANCE FROM BAL_SHET WHERE ACCT_NUM = "2130" +
   AND ACCT_NUM = "2710"
COMPUTE VINSUR AS SUM BALANCE FROM BAL_SHET WHERE ACCT_NUM = "2530" +
   AND ACCT_NUM = "2140"
COMPUTE VSHTERM AS SUM BALANCE FROM BAL_SHET WHERE ACCT_NUM = "2510"
COMPUTE VTOTLIAB AS SUM BALANCE FROM BAL_SHET WHERE ACCT_NUM >= "2100" +
   AND ACCT_NUM <= "2710"
PRINT OPSRPT2
NEWPAGE
CLEAR VAP
CLEAR VMISCL
CLEAR VINSUR
CLEAR VSHTERM

*(-- printing the first of the operating income/expenses sheets)
*(-- computing income and expense values needed for the report)
SET VARIABLE VRETAIN TO .TOTASET - .TOTLIAB
COMPUTE VFLTSUP AS SUM BALANCE FROM EARNINGS WHERE ACCT_NUM = "4941"
COMPUTE VFLTCOST AS SUM BALANCE FROM EARNINGS WHERE ACCT_NUM = "5941"
COMPUTE VRENT AS SUM BALANCE FROM EARNINGS WHERE ACCT_NUM = "4951"
COMPUTE VCFI AS SUM BALANCE FROM EARNINGS WHERE ACCT_NUM = "4952"
COMPUTE VADMIN AS SUM BALANCE FROM EARNINGS WHERE ACCT_NUM = "7110"
COMPUTE VMAINT AS SUM BALANCE FROM EARNINGS WHERE ACCT_NUM >= "7120"
COMPUTE VACCOST AS SUM BALANCE FROM EARNINGS WHERE ACCT_NUM = "6950" +
   AND ACCT_NUM = "7130"
COMPUTE VGAS AS SUM BALANCE FROM EARNINGS WHERE ACCT_NUM = "7701" +
   AND ACCT_NUM = "7703"
COMPUTE VOIL AS SUM BALANCE FROM EARNINGS WHERE ACCT_NUM = "7702"
COMPUTE VMAIN1 AS SUM BALANCE FROM EARNINGS WHERE ACCT_NUM = "7121"
COMPUTE VMAIN2 AS SUM BALANCE FROM EARNINGS WHERE ACCT_NUM >= "7740" +
   AND ACCT_NUM <= "7742"
SET VARIABLE VACMAINT TO .VMAIN1 + .VMAIN2
COMPUTE VUTIL AS SUM BALANCE FROM EARNINGS WHERE ACCT_NUM = "7511" +
   AND ACCT_NUM = "7513"
COMPUTE VTEL AS SUM BALANCE FROM EARNINGS WHERE ACCT_NUM = "7520" +
   AND ACCT_NUM = "7521"
COMPUTE VSUP AS SUM BALANCE FROM EARNINGS WHERE ACCT_NUM = "7710"
COMPUTE VINSCOST AS SUM BALANCE FROM EARNINGS WHERE ACCT_NUM >= "7810" +
   AND ACCT_NUM <= "7812"
COMPUTE VINC1 AS SUM BALANCE FROM EARNINGS WHERE ACCT_NUM >= "8100"
   AND ACCT_NUM <= "8192"
```

```
COMPUTE VINC2 AS SUM BALANCE FROM EARNINGS WHERE ACCT_NUM = "8301"
SET VARIABLE VOTHINC TO .VINC1 + .VINC2
COMPUTE VINTST AS SUM BALANCE FROM EARNINGS WHERE ACCT_NUM = "8300"
COMPUTE VMISC1 AS SUM BALANCE FROM EARNINGS WHERE ACCT_NUM = "7930" +
   AND ACCT_NUM = "7820"
COMPUTE VMISC2 AS SUM BALANCE FROM EARNINGS WHERE ACCT_NUM = "7522" +
   AND ACCT_NUM = "7512"
COMPUTE VMISC3 AS SUM BALANCE FROM EARNINGS WHERE ACCT_NUM = "7711"
SET VARIABLE VMISC4 TO .VMISC1 + .VMISC2
SET VARIABLE VMSEXP TO .VMISC3 + .VMISC4
COMPUTE VINTEXP AS SUM BALANCE FROM EARNINGS WHERE ACCT_NUM = "7712"
COMPUTE VBADDBT AS SUM BALANCE FROM EARNINGS WHERE ACCT_NUM = "9255"

PRINT OPSRPT3
NEWPAGE
*(-- printing the last of the operating income/expense sheets)
*(-- print routine uses the values computed above also)

COMPUTE VTOTINC1 AS SUM BALANCE FROM EARNINGS WHERE ACCT_NUM >= "4941" +
   AND ACCT_NUM <= "4952"
COMPUTE VTOTINC2 AS SUM BALANCE FROM EARNINGS WHERE ACCT_NUM >= "8100" +
   AND ACCT_NUM <= "8301"
SET VARIABLE VTOTINC TO .VTOTINC1 + .VTOTINC2
COMPUTE VTOTEXP1 AS SUM BALANCE FROM EARNINGS WHERE ACCT_NUM >= "5941" +
   AND ACCT_NUM <= "7930"
SET VARIABLE VTOTEXP TO .VBADDBT + .VTOTEXP1
SET VARIABLE VNETINC TO .TOTINC - .VTOTEXP

PRINT OPSRPT4
NEWPAGE

*(-- prepare netincome or netloss for the statement of net worth)
IF VNETINC < 0 THEN
   SET VARIABLE VNETINC TO .VNETINC X -1
   SET VARIABLE VNETLOSS TO .VNETINC
   SET VARIABLE VNETINC TO 0
ENDIF
IF VNETINC >= 0 THEN
   SET VARIABLE VNETLOSS TO 0
ENDIF

*(-- printing the statement of net worth using data computed above)
PRINT OPSRPT5
NEWPAGE

CLEAR VTOTASET
CLEAR VTOTLIAB
CLEAR VTOTINC
CLEAR VTOTINC1
CLEAR VTOTINC2
CLEAR VTOTEXP
CLEAR VTOTEXP1
CLEAR VNETINC
CLEAR VNETLOSS
CLEAR VMISC1
CLEAR VMISC2
CLEAR VMISC3
CLEAR VMISC4
CLEAR VFLTSUP
CLEAR VFLTCOST
CLEAR VRENT
CLEAR VCFI
CLEAR VADMIN
CLEAR VMAINT
CLEAR VACCOST
CLEAR VGAS
CLEAR VOIL
CLEAR VUTIL
CLEAR VTEL
CLEAR VSUP
CLEAR VINSCOST
CLEAR VOTHINC
CLEAR VINTST
CLEAR VMSEXP
CLEAR VINTEXP
CLEAR VBADDBT
*(--  return control to screen and user )
OUTPUT SCREEN

NEWPAGE
WRITE "OPERATING STATEMENTS ARE COMPLETE." AT 7,23
WRITE "PRESS ANY KEY TO RETURN TO MENU." AT 10,25
PAUSE

LABEL OPSMTEND
SET MESSAGES ON
SET ERROR MESSAGES ON
RETURN
*(* END OF OP_STMT.CMD )
$COMMAND
INS_SUM
*(*******************************************************************
PROGRAM:       INS_SUM.CMD
AUTHOR:        J. M. GRAHAM
DATE:          JAN 1987      VER 1.2
DESCRIPTION:   THIS MODULE PREPARES THE SUMMARY OF INSURANCE ON ALL
               AIRCRAFT IN THE CURRENT INVENTORY.  THIS REPORT IS
               ONE OF THE REQUIRED ANNUAL REPORT INPUTS.

TABLES USED:   A/C_REC
FORMS USED:    NONE
REPORT USED:   INS_SUM                                       (17)
*******************************************************************)

SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

NEWPAGE
WRITE "THE INSURANCE SUMMARY SHOULD BE PRINTED AS NEEDED." AT 5,10
WRITE "DO YOU WANT TO PROCEED? (Y/N) - " AT 10,17
FILLIN YOURANS USING " " AT 10,50
IF YOURANS = N THEN
   GOTO INSUMEND
ENDIF

NEWPAGE
SET VARIABLE YR1 TEXT
SET VARIABLE YR2 TEXT
FILLIN YR1 USING "INSURANCE REPORT COVERING YEAR 19  -" AT 5,10
FILLIN YR2 USING "ENDING YEAR 19  -"  AT 7,10
WRITE "YOU WILL HAVE TO FILL-IN THE NUMBER DAYS PER PLANE"  AT 10,5
WRITE "AFTER THE INSURANCE SUMMARY IS PRINTED"  AT 11,10
WRITE "ENSURE PRINTER IS READY, THEN PRESS ANY KEY TO CONTINUE." AT 15,5
PAUSE

NEWPAGE
WRITE "PRINTING THE INSURANCE SUMMARY"  AT 10,20
WRITE "THIS WILL TAKE A LITTLE TIME, WHY NOT TAKE FIVE ON ME." AT 15,8

*(-- Print insurance summary )
OUTPUT SCREEN WITH PRINTER
*(-- giving the manager the material for rough work )
SELECT A/C_NUM TYPE YEAR MODEL PASSEAT=3 NO_ENG=3 START STOP +
   HULLVAL=6 HULLPREM=6 LIABPREM=6 TOTPREM=7 FROM A/C_REC
NEWPAGE
*( PRINT INS_SUM )
OUTPUT SCREEN
```

```
NEWPAGE
WRITE "INSURANCE SUMMARY IS COMPLETE." AT 7,10
WRITE "PRESS ANY KEY TO RETURN TO MENU." AT 10,15
PAUSE

LABEL INSUMEND
SET MESSAGES ON
SET ERROR MESSAGES ON
RETURN
*(* END OF INS_SUM.CMD)

$COMMAND
AGED_AR
*(*********************************************************************
PROGRAM:      AGED_AR.CMD
AUTHOR:       J. M. GRAHAM
DATE:         JAN 1987    VER 1.3
DESCRIPTION:  THIS MODULE PREPARES AND PRINTS THE SUMMARY OF AGED
              ACCOUNTS RECEIVABLE.

TABLES USED:  MEM_REC, ACCT_REC
FORMS USED:   NONE
REPORT USED:  AGED_AR                                          (17)
*********************************************************************)

SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

NEWPAGE
WRITE "AGED ACCOUNTS RECEIVABLE REPORT MAY BE PRINTED ANY TIME."AT 5,5
WRITE "HOWEVER, THE ACCOUNTS ARE ONLY UPDATED MONTHLY AND MAY"  AT 7,5
WRITE "NOT BE CURRENT UNLESS YOU HAVE COMPLETED THE END-OF-THE"  AT 8,5
WRITE "MONTH POSTING.  RECOMMEND YOU PRINT REPORT AFTER YOU HAVE"AT 9,5
WRITE "COMPLETED THE MONTH'S BILLING AND POSTING."  AT 10,10
WRITE "DO YOU WANT TO PROCEED? (Y/N) - " AT 15,17
FILLIN YOURANS USING " " AT 15,50
IF YOURANS = N THEN
   GOTO AGEDEND
ENDIF

NEWPAGE
WRITE "ENSURE PRINTER IS READY, THEN PRESS ANY KEY TO CONTINUE."AT10,5
PAUSE

NEWPAGE
WRITE "PRINTING THE AGED ACCOUNTS RECEIVABLE REPORT"  AT 10,10
WRITE "THIS WILL TAKE A LITTLE TIME, WHY NOT TAKE FIVE ON ME." AT 15,15

*(-- Print aged accounts receivalbe report)
OUTPUT PRINTER
PRINT AGED_AR
OUTPUT SCREEN

NEWPAGE
WRITE "AGED ACCOUNTS RECEIVABLE REPORT COMPLETE." AT 7,10
WRITE "PRESS ANY KEY TO RETURN TO MENU." AT 10,15
PAUSE

LABEL AGEDEND
SET MESSAGES ON
SET ERROR MESSAGES ON
RETURN
*(* END OF AGED_AR.CMD *)

$COMMAND
PRT_BAL
*(*********************************************************************
PROGRAM:       PRT_BAL.CMD
AUTHOR:        J. M. GRAHAM
DATE:          JAN 1987     VER 1.3
DESCRIPTION:   THIS MODULE TAKES THE INFORMATION LOCATED IN THE BALANCE
               SHEET TABLE AND PREPARES AND PRINTS A CURRENT BALANCE
               SHEET.

TABLES USED:   BAL_SHET
FORMS USED:    NONE
REPORT USED:   BALSHET                                          (16)
**********************************************************************)

SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

NEWPAGE
WRITE "THE BALANCE SHEET SHOULD BE PRINTED AT MONTH'S END." AT 5,5
WRITE "ALL ACCOUNT POSTING IS CONDUCTED MONTHLY.  YOUR REPORT"  AT 7,5
WRITE"MAY NOT BE ACCURATE UNLESS YOU HAVE COMPLETED THIS MONTH'S"AT 8,5
WRITE "BILLING AND POSTING."  AT 9,5
WRITE "DO YOU WANT TO PROCEED? (Y/N) - " AT 15,17
FILLIN YOURANS USING " " AT 15,50
IF YOURANS = N THEN
   GOTO BALEND
ENDIF

NEWPAGE
WRITE "ENSURE PRINTER IS READY, THEN PRESS ANY KEY TO CONTINUE."AT 10,5
PAUSE

NEWPAGE
WRITE "PRINTING THE CURRENT BALANCE SHEET AS OF LAST UPDATE"  AT 10,5
WRITE "THIS WILL TAKE A LITTLE TIME, WHY NOT TAKE FIVE ON ME." AT 15,8

*(-- Print current balance sheet as of last database update)
SET DATE MMM-DD-YYYY

OUTPUT PRINTER
SET VARIABLE NETWRTH DOLLAR
SET VARIABLE GRANDTOT DOLLAR

COMPUTE TOTASET AS SUM BALANCE FROM BAL_SHET WHERE ACCT_NUM LT "2100"
COMPUTE TOTLIAB AS SUM BALANCE FROM BAL_SHET WHERE ACCT_NUM GE "2100"
SET VARIABLE NETWRTH TO .TOTASET - .TOTLIAB
SET VARIABLE GRANDTOT TO .TOTLIAB + .NETWRTH

PROJECT TEMP1 FROM BAL_SHET USING ALL

DELETE ROWS FROM BAL_SHET WHERE ACCT_NUM GE "2100"

PRINT BALSHET1
APPEND TEMP1 TO BAL_SHET WHERE ACCT_NUM GE "2100"
DELETE ROWS FROM BAL_SHET WHERE ACCT_NUM LT "2100"
PRINT BALSHET2

APPEND TEMP1 TO BAL_SHET WHERE ACCT_NUM LT "2100"
REMOVE TEMP1

SET DATE MM/DD/YY
CLEAR TOTASET
CLEAR TOTLIAB
CLEAR NETWRTH
CLEAR GRANDTOT

OUTPUT SCREEN

NEWPAGE
WRITE "BALANCE SHEET IS COMPLETE." AT 7,10
WRITE "REMEMBER TO BACK-UP AND PACK YOUR DATABASE ONCE YOU"  AT 10,15
WRITE "HAVE COMPLETED PRINTING ALL MONTHLY REPORTS"  AT 12,19
WRITE "PRESS ANY KEY TO RETURN TO MENU." AT 20,15
PAUSE

LABEL BALEND
SET MESSAGES ON
SET ERROR MESSAGES ON
RETURN
*(* END OF PRT_BAL.CMD *)


$COMMAND
PRT_INC
*(*********************************************************************
PROGRAM:      PRT_INC.CMD
AUTHOR:       J. M. GRAHAM
DATE:         JAN 1987    VER 1.3
DESCRIPTION:  THIS MODULE PREPARES AND PRINTS THE INCOME STATEMENT
              UTILIZING THE LAST MONTHLY UPDATE TO THE EARNINGS
              TABLE.  THIS DOES NOT ACCOUNT FOR ENTRIES MADE AFTER
              MONTHLY UPDATES...WILL BE ON NEXT MONTH'S TOTALS.
              NORMAL INCOME STATEMENT PRINTED YEARLY.

TABLES USED:  EARNINGS
FORMS USED:   NONE
REPORT USED:  INCSTMT                                   (19)
*********************************************************************)

SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

NEWPAGE
WRITE "THE INCOME STATEMENT SHOULD BE PRINTED AT YEAR'S END." AT 5,5
WRITE "ALL ACCOUNTS ARE UPDATED AND POSTED MONTHLY.  YOUR REPORT"AT 7,5
WRITE "WILL BE ACCURATE AS OF THE LAST MONTH POSTING UNLESS YOU"AT 8,5
WRITE "COMPLETED THE LAST MONTH OF THE YEAR POSTING."  AT 8,10
WRITE "FOR FISCAL YEAR REPORTS PRINT AFTER SEPTEMBER'S POSTINGS"AT 10,5
WRITE "DO YOU WANT TO PROCEED? (Y/N) - " AT 15,17
FILLIN YOURANS USING " " AT 15,50
IF YOURANS = N THEN
   GOTO INCEND
ENDIF

NEWPAGE
FILLIN VFY USING "REPORT FOR FY - "  AT 5,25
FILLIN VMONTH USING "REPORT ENDING MONTH - "  AT 7,25
FILLIN VYEAR USING "OF YEAR - "  AT 9,35

WRITE "ENSURE PRINTER IS READY." AT 13,27
WRITE "PRESS ANY KEY TO BEGIN PRINTING OF INCOME STATEMENT"  AT 18,14
PAUSE

NEWPAGE
WRITE "PRINTING THE INCOME STATEMENT AS OF LAST UPDATE"  AT 10,16
WRITE "THIS WILL TAKE A LITTLE TIME, WHY NOT TAKE FIVE ON ME." AT 15,13

*(-- Print current income statement as of last database update)

SET VARIABLE VTOTINC DOLLAR
SET VARIABLE VTOTEXP DOLLAR
SET VARIABLE VNET DOLLAR
*(-- saves the current earnings table values )
PROJECT TEMP1 FROM EARNINGS USING ALL

*(--  separate income from earnings table to compute income)
DELETE ROWS FROM EARNINGS WHERE ACCT_NUM >= "5941" +
   AND ACCT_NUM <= "7930"
```

```
DELETE ROWS FROM EARNINGS WHERE ACCT_NUM = "9255"
COMPUTE VTOTINC AS SUM BALANCE FROM EARNINGS
OUTPUT PRINTER
PRINT INCSTMT1

*(-- separate expenses from earnings table to compute expense only)
APPEND TEMP1 TO EARNINGS WHERE ACCT_NUM >= "5941" +
   AND ACCT_NUM <= "7930"
APPEND TEMP1 TO EARNINGS WHERE ACCT_NUM = "9255"
*(-- remove all income from table to print expenses)
DELETE ROWS FROM EARNINGS WHERE ACCT_NUM >= "4941" +
   AND ACCT_NUM <= "4952"
DELETE ROWS FROM EARNINGS WHERE ACCT_NUM >= "8100" +
   AND ACCT_NUM <= "8301"
COMPUTE VTOTEXP AS SUM BALANCE FROM EARNINGS
SET VARIABLE VNET TO .VTOTINC - .VTOTEXP
PRINT INCSTMT2
*(-- delete working values and restore earnings table as was)
DELETE ROWS FROM EARNINGS WHERE ACCT_NUM EXISTS
APPEND TEMP1 TO EARNINGS
REMOVE TEMP1

CLEAR VTOTINC
CLEAR VTOTEXP
CLEAR VNET
OUTPUT SCREEN

NEWPAGE
WRITE "INCOME STATEMENT IS COMPLETE." AT 7,25
WRITE "PRESS ANY KEY TO RETURN TO MENU." AT 10,23
PAUSE

LABEL INCEND
SET MESSAGES ON
SET ERROR MESSAGES ON
RETURN
*(* END PRT_INC.CMD *)
```

APPENDIX H
CLUB_END SOURCE CODE

```
*(**********************************************************************)
*(*                    MONTEREY NAVY FLYING CLUB                       *)
*(*                 MANAGEMENT INFORMATION SYSTEM                      *)
*(*                    PROTOTYPE VERSION 2.0                           *)
*(*                                                                    *)
*(*           END-OF-THE-MONTH TRANSACTIONS APPLICATION                *)
*(*                                                                    *)
*(*                   LCDR. JAMES M. GRAHAM, USN                       *)
*(*                         JANUARY 1987                          (07) *)
*(**********************************************************************)

$COMMAND
CLUB_END
SET MESSAGE OFF
OPEN FLYCLUB
SET ERROR MESSAGE OFF
SET VAR PICK1  INT
LABEL STARTAPP
  NEWPAGE
  CHOOSE PICK1  FROM END_MAIN IN CLUB_END.APX
  IF PICK1  EQ 0 THEN
    GOTO ENDAPP
  ENDIF
  IF PICK1  EQ            1 THEN
    RUN REOCCUR  IN CLUB_END.APX
    GOTO STARTAPP
  ENDIF
  IF PICK1  EQ            2 THEN
    SET VAR PICK2  INT
    SET VAR LEVEL2 INT
    SET VAR LEVEL2 TO 0
    WHILE LEVEL2 EQ 0  THEN
      NEWPAGE
      CHOOSE PICK2  FROM BILLING  IN CLUB_END.APX
      IF PICK2  EQ 0 THEN
        BREAK
      ENDIF
      IF PICK2  EQ            1 THEN
        RUN MEMSTMT  IN CLUB_END.APX
      ENDIF
      IF PICK2  EQ            2 THEN
        RUN INSTSTMT IN CLUB_END.APX
      ENDIF
      IF PICK2  EQ            3 THEN
        RUN LES_STMT IN CLUB_END.APX
      ENDIF
      IF PICK2  EQ            4 THEN
        BREAK
      ENDIF
    ENDWHILE
    CLEAR LEVEL2
    CLEAR PICK2
    GOTO STARTAPP
  ENDIF
  IF PICK1  EQ            3 THEN
    RUN END_MON  IN CLUB_END.APX
    GOTO STARTAPP
  ENDIF
  IF PICK1  EQ            4 THEN
    RUN BACKPACK IN CLUB_END.APX
    GOTO STARTAPP
  ENDIF
  IF PICK1  EQ             5 THEN
    GOTO ENDAPP
  ENDIF
  GOTO STARTAPP
LABEL ENDAPP
CLEAR PICK1
RETURN
$MENU
END_MAIN
COLUMN END-OF-THE-MONTH TRANSACTIONS
POSTING REOCCURRING CHARGES
POST AND PREPARE MONTHLY BILLINGS
POST MONTHLY JOURNAL ENTRIES
BACK UP AND PACK THE DATABASE
RETURN TO THE MNFC MAIN MENU
$MENU
BILLING
COLUMN POST AND PREPARE MONTHLY BILLINGS
POST AND PRINT MEMBER STATEMENTS
POST AND PRINT INSTRUCTOR STATEMENTS
POST AND PRINT LESSOR STATEMENTS
RETURN TO END-OF-THE-MONTH MENU
$COMMAND
REOCCUR
*(********************************************************************
PROGRAM:      REOCCUR.CMD
AUTHOR:       J. M. GRAHAM
DATE:         JAN 1987    VER. 1.3
DESCRIPTION:  THIS MODULE POST REOCCURRING CHARGES TO THE MEMBER'S
              CHARGE ACCOUNT.  THE MANAGER WILL NOTE WHETHER BOTH
              THE MONTHLY DUES AND THE MEMBER MEETING CHARGES ARE TO
              BE POSTED OR JUST ONE TO BE POSTED TO EACH MEMBER.

TABLES USED:  MEM_REC, MEM_CHG
FORMS USED:   NONE                                             (16)
*********************************************************************)

SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

*(-- screen memu to ask if both charges are required )
NEWPAGE
WRITE "POSTING REOCCURRING CHARGES"  AT 5,25
WRITE "---------------------------"  AT 6,25
WRITE "ENTER:"  AT 9,10
WRITE "1 - POST ONLY MONTHLY MEMBERSHIP DUES"  AT 11,10
WRITE "2 - POST BOTH MONTHLY DUES AND MEETING CHARGES"  AT 13,10
WRITE "3 - EXIT WITH NO POSTING"  AT 15,10
WRITE "CHOICE:"  AT 17,10
FILLIN YOURCH USING " "  AT 17,20
IF YOURCH = 1 THEN
   GOTO POSTDUES *(-- call dues posting submodule )
ENDIF
IF YOURCH = 2 THEN
   GOTO POSTBOTH *(-- call subroutine to post both charges )
ENDIF
IF YOURCH = 3 THEN
   GOTO REOCEND  *(-- exit routine)
ENDIF

*(--  subroutine that determines and post correct dues to member)
LABEL POSTDUES
NEWPAGE
*(-- screen display to tell user operations have begun )
WRITE "Mate, I would like cream and sugar in my coffee" AT 5,15
WRITE "What do you want in YOURS?" AT 7,20
WRITE "Why not take that well deserved coffee break while I work"AT 9,5
WRITE "on the reoccurring charges" AT 10,5
WRITE "PRESS ANY KEY FOR ME TO BEGIN WORKING" AT 12,20
PAUSE
SET VARIABLE VACCTNUM TO "8100"
SET VARIABLE VSINGMBR TO "SM"
SET VARIABLE VCOUPLE TO "MC"
SET VARIABLE VMBRDEP TO "DM"
SET VARIABLE VTX:DATE TO .#DATE
SET VARIABLE VDUECHG TO DOLLAR
SET VARIABLE VDUEINC TO DOLLAR
SET VARIABLE VACTIVE TO "A"

*(-- charge single member dues )
SET VARIABLE VDUECHG TO 7.50
*(-- compute single member dues and post to income table)
COMPUTE VSINGCT AS COUNT DUE_STAT FROM MEM_REC WHERE MEM_STAT EQ +
   .VACTIVE AND DUE_STAT EQ .VSINGMBR
IF VSINGCT EXITS THEN
   SET VARIABLE VNAME TO "MONTHLY SINGMBR DUES"
   SET VARIABLE VDUEINC TO .VDUECHG X .VSINGCT
   LOAD INCOME
     .VACCTNUM .VNAME .VDUEINC .VTX:DATE P
   END
   CLEAR VDUEINC
ENDIF  *(-- end of posting income from single dues)

SET POINTER #1 STATUS1 FOR MEM_REC WHERE MEM_STAT EQ .VACTIVE +
    AND DUE_STAT EQ .VSINGMBR
WHILE STATUS1 = 0 THEN
    SET VARIABLE VMEMNUM TO MEM_NUM IN #1
    LOAD MEM_CHG USING MEM_NUM TRANDATE CHARGES DESCRIPT
        .VMEMNUM .VTX:DATE .VDUECHG DUES
    END
ENDWHILE  *(-- end of single due charges )
*(-- married couple due charges )
SET VARIABLE VDUECHG TO 22.50
*(-- compute married members dues and post to income table)
COMPUTE VCOUPCT AS COUNT DUE_STAT FROM MEM_REC WHERE MEM_STAT EQ +
   .VACTIVE AND DUE_STAT EQ .VCOUPLE
IF VCOUPCT EXITS THEN
   SET VARIABLE VNAME TO "MONTHLY COUPLE DUES"
   SET VARIABLE VDUEINC TO .VDUECHG X .VSINGCT
   LOAD INCOME
     .VACCTNUM .VNAME .VDUEINC .VTX:DATE P
   END
   CLEAR VDUEINC
ENDIF  *(-- end of posting income from married member dues)

SET POINTER #1 STATUS1 FOR MEM_REC WHERE MEM_STAT EQ .VACTIVE +
    AND DUE_STAT EQ .VCOUPLE
WHILE STATUS1 = 0 THEN
    SET VARIABLE VMEMNUM TO MEM_NUM IN #1
    LOAD MEM_CHG USING MEM_NUM TRANDATE CHARGES DESCRIPT
        .VMEMNUM .VTX:DATE .VDUECHG DUES
    END
ENDWHILE  *(-- end of married couple due charges )
*(-- dependent member due charges )
SET VARIABLE VDUECHG TO 15
*(-- compute dependent member dues and post to income table)
COMPUTE VDEPCT AS COUNT DUE_STAT FROM MEM_REC WHERE MEM_STAT EQ +
   .VACTIVE AND DUE_STAT EQ .VMBRDEP
IF VDEPCT EXITS THEN
   SET VARIABLE VNAME TO "MONTHLY DEPENDENT MEMBER DUES"
   SET VARIABLE VDUEINC TO .VDUECHG X .VSINGCT
   LOAD INCOME
     .VACCTNUM .VNAME .VDUEINC .VTX:DATE P
   END
   CLEAR VDUEINC
ENDIF  *(-- end of posting income from dependent member dues)
```

```
SET POINTER #1 STATUS1 FOR MEM_REC WHERE MEM_STAT EQ .VACTIVE +
    AND DUE_STAT EQ .VMBRDEP
WHILE STATUS1 = 0 THEN
    SET VARIABLE VMEMNUM TO MEM_NUM IN #1
    LOAD MEM_CHG USING MEM_NUM TRANDATE CHARGES DESCRIPT
        .VMEMNUM .VTX:DATE .VDUECHG DUES
    END
ENDWHILE *( -- end of dependent member due charges )
NEWPAGE
WRITE "POSTING OF MEMBER DUES IS COMPLETED"  AT 10,10
WRITE "PRESS ANY KEY TO RETURN TO MEMU"  AT 15,10
PAUSE
GOTO REOCEND

LABEL POSTBOTH
*(-- subroutine posting both member dues and meeting charges )
NEWPAGE
SET VARIABLE VMTGCHG TO DOLLAR
FILLIN VMTGCHG USING "AMOUNT OF MEETING CHARGE IS - "  AT 10,10
WRITE "WHY NOT TAKE THAT WELL DESERVED COFFEE BREAK"  AT 15,10
WRITE "AND LET ME WORK ON THESE REOCCURRING CHARGES"  AT 16,10
WRITE "PRESS ANY KEY FOR ME TO BEGIN"  AT 20,20
PAUSE
SET VARIABLE VACCTNUM TO "8190"
SET VARIABLE VSINGMBR TO "SM"
SET VARIABLE VCOUPLE TO "MC"
SET VARIABLE VMBRDEP TO "DM"
SET VARIABLE VTX:DATE TO .#DATE
SET VARIABLE VDUECHG TO DOLLAR
SET VARIABLE VACTIVE TO "A"
SET VARIABLE VMTGFEE TO "MEETING CHARGE"
*(-- post member meeting charges to all active members )
*(-- compute total income from meeting charges and post to income)
COMPUTE VMBRCT AS COUNT MEM_STAT FROM MEM_REC WHERE MEM_STAT +
   EQ .VACTIVE
SET VARIABLE VMTGAMT TO DOLLAR
SET VARIABLE VMTGAMT TO .VMBRCT X .VMTCHG
LOAD INCOME
   .VACCTNUM .VTX:DATE .VMTGAMT .VMTGFEE
END

SET POINTER #1 STATUS1 FOR MEM_REC WHERE MEM_STAT EQ .VACTIVE
WHILE STATUS1 = 0 THEN
    SET VARIABLE VMEMNUM TO MEM_NUM IN #1
    LOAD MEM_CHG USING MEM_NUM TRANDATE CHARGES DESCRIPT
        .VMEMNUM .VTX:DATE .VMTGCHG .VMTGFEE
    END
ENDWHILE
*(-- charge single member dues )
SET VARIABLE VDUECHG TO 7.50
*(-- compute single member dues and post to income table)
COMPUTE VSINGCT AS COUNT DUE_STAT FROM MEM_REC WHERE MEM_STAT EQ +
   .VACTIVE AND DUE_STAT EQ .VSINGMBR
IF VSINGCT EXITS THEN
   SET VARIABLE VNAME TO "MONTHLY SINGMBR DUES"
   SET VARIABLE VDUEINC TO .VDUECHG X .VSINGCT
   LOAD INCOME
      .VACCTNUM .VNAME .VDUEINC .VTX:DATE P
   END
   CLEAR VDUEINC
ENDIF  *(-- end of posting income from single dues)

SET POINTER #1 STATUS1 FOR MEM_REC WHERE MEM_STAT EQ .VACTIVE +
    AND DUE_STAT EQ .VSINGMBR
WHILE STATUS1 = 0 THEN
    SET VARIABLE VMEMNUM TO MEM_NUM IN #1
    LOAD MEM_CHG USING MEM_NUM TRANDATE CHARGES DESCRIPT
        .VMEMNUM .VTX:DATE .VDUECHG DUES
    END
ENDWHILE  *(-- end of single due charges )
*(-- married couple due charges )
SET VARIABLE VDUECHG TO 22.50
*(-- compute married members dues and post to income table)
COMPUTE VCOUPCT AS COUNT DUE_STAT FROM MEM_REC WHERE MEM_STAT EQ +
   .VACTIVE AND DUE_STAT EQ .VCOUPLE
IF VCOUPCT EXITS THEN
   SET VARIABLE VNAME TO "MONTHLY COUPLE DUES"
   SET VARIABLE VDUEINC TO .VDUECHG X .VSINGCT
   LOAD INCOME
     .VACCTNUM .VNAME .VDUEINC .VTX:DATE P
   END
   CLEAR VDUEINC
ENDIF  *(-- end of posting income from married member dues)

SET POINTER #1 STATUS1 FOR MEM_REC WHERE MEM_STAT EQ .VACTIVE +
    AND DUE_STAT EQ .VCOUPLE
WHILE STATUS1 = 0 THEN
    SET VARIABLE VMEMNUM TO MEM_NUM IN #1
    LOAD MEM_CHG USING MEM_NUM TRANDATE CHARGES DESCRIPT
        .VMEMNUM .VTX:DATE .VDUECHG DUES
    END
ENDWHILE  *(-- end of married couple due charges )
*(-- dependent member due charges )
SET VARIABLE VDUECHG TO 15
*(-- compute dependent member dues and post to income table)
COMPUTE VDEPCT AS COUNT DUE_STAT FROM MEM_REC WHERE MEM_STAT EQ +
   .VACTIVE AND DUE_STAT EQ .VMBRDEP
IF VDEPCT EXITS THEN
   SET VARIABLE VNAME TO "MONTHLY DEPENDENT MEMBER DUES"
   SET VARIABLE VDUEINC TO .VDUECHG X .VSINGCT
   LOAD INCOME
     .VACCTNUM .VNAME .VDUEINC .VTX:DATE P
   END
   CLEAR VDUEINC
ENDIF  *(-- end of posting income from dependent member dues)

SET POINTER #1 STATUS1 FOR MEM_REC WHERE MEM_STAT EQ .VACTIVE +
    AND DUE_STAT EQ .VMBRDEP
WHILE STATUS1 = 0 THEN
    SET VARIABLE VMEMNUM TO MEM_NUM IN #1
    LOAD MEM_CHG USING MEM_NUM TRANDATE CHARGES DESCRIPT
        .VMEMNUM .VTX:DATE .VDUECHG DUES
    END
ENDWHILE *( -- end of dependent member due charges )
NEWPAGE
WRITE"POSTING OF MEMBERS' DUES AND MEETING CHARGES IS COMPLETED"AT 10,10
WRITE "PRESS ANY KEY TO RETURN TO THE MENU"  AT 20,20
PAUSE

LABEL REOCEND
SET MESSAGES ON
SET ERROR MESSAGES ON
RETURN
*(* END OF REOCCUR.CMD )

$COMMAND
END_MON
*(*****************************************************************
PROGRAM:      END_MON.CMD
AUTHOR:       J. M. GRAHAM
DATE:         JAN 1987     VER 1.5
DESCRIPTION:  THIS MODULE POST THE MONTH JOURNAL ENTRIES OF MOST
              DATABASE TABLES AND PLACES ACCOUNT TOTALS IN THE
              GENERAL PURPOSE TABLES.  THEN THE JOURNAL ENTRIES ARE
              PLACED IN THEIR ASSOCIATED HISTORICAL TABLE AND THE
              TABLES ARE PREPARED FOR THE NEXT MONTH'S TRANSACTIONS.
              SOME TABLES ARE POSTED AND THEN PLACED IN HISTORICAL
              TABLES DURING EXECUTION OF ANOTHER MODULE (EG. MEMBER
              CHARGES AND PAYMENTS TABLES POSTED DURING PRINTING OF
              MEMBER STATEMENTS.)

              THEREFORE, THE LESSOR'S STATEMENTS, THE AIRCRAFT
              INVENTORY AND STATUS REPORT, THE MEMBER STATEMENTS, AND
              INSTRUCTOR STATEMENTS MUST BE POSTED PRIOR TO THIS
              MODULE'S OPERATION OR THE POSTING OF TABLES WILL BE
              INACCURATE.

              AFTER THE POSTINGS ARE COMPLETED THE USER WILL BE
              DIRECTED TO PERFORM A DATABASE BACK-UP AND REPACK
              INORDER FOR THE DATABASE TO PERFORM MORE EFFICIENTLY.

TABLES USED:  INCOME, EXPENSE, EARNINGS, INC_HIST, EXP_HIST, BAL_SHET,
              BAL_JOUR, ACCT_PAY, AP_CHG, AP_PAID, APCGHIST, APPDHIST,
              INS_REC, MEM_FLT, FLT_REC, FLT_HIST, A/CMAINT, MAINHIST,
              ACCT_REC, CAP_ASET, MBR_SUM
FORMS USED:   NONE                                        (20)
*****************************************************************)

SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

*(-- system warning of necessary modules to run prior)
NEWPAGE
WRITE "!!!!! WARNING !!!!!"  AT 3,30
WRITE "ENSURE YOU HAVE COMPLETED:"  AT 8,5
WRITE "LESSOR'S STATEMENT"  AT 10,15
WRITE "THE AIRCRAFT INVENTORY AND STATUS REPORT"  AT 12,15
WRITE "THE MEMBERS' STATEMENTS"  AT 14,15
WRITE "THE INSTRUCTOR STATEMENT"  AT 16,15
WRITE "BEFORE YOU EXECUTE THIS MODULE"  AT 18,5
WRITE "YOU NEED TWO FORMATED DISKS ALSO BEFORE PROCEEDING" AT 20,15
WRITE "DO YOU DESIRE ME TO BEGIN WORKING?(Y/N)"  AT 22,20
FILLIN YOURANS USING " "  AT 22,65
IF YOURANS = N THEN
   NEWPAGE
   WRITE "EXITING END-OF-THE-MONTH POSTING WITHOUT POSTING"  AT 10,15
   WRITE "PRESS ANY KEY TO RETURN TO THE MENU"  AT 15,22
   PAUSE
   GOTO MONTHEND
ENDIF

*(-- screen display to tell user operations have begun )
NEWPAGE
WRITE "Mate, I would like cream and sugar in my coffee" AT 5,15
WRITE "What do you want in YOURS?" AT 7,25
WRITE "Why not take that well deserved coffee break while I work" AT 9,5
WRITE "on the End-of-the-Month Transactions" AT 10,5

*(-- global variables)
SET VARIABLE VPOSTDAT TO .#DATE
SET VARIABLE VAMT DOLLAR
SET VARIABLE VMINAMT DOLLAR
SET VARIABLE VCURBAL DOLLAR
SET VARIABLE VPLUS TO "P"
SET VARIABLE VMINUS TO "M"
SET VARIABLE VHRS REAL
SET VARIABLE VCOUNT INTEGER
*(* POSTING ACCOUNTS PAYABLE ACCOUNTS)
*(-- computing all charges and payments to accounts)
APPEND AP_CHG TO APTEMP
APPEND AP_PAID TO APTEMP
CHANGE CURRBAL TO 0 IN ACCT_PAY WHERE CURRBAL FAILS
SET POINTER #1 STATUS1 FOR ACCT_PAY
  WHILE STATUS1 = 0 THEN
```

```
    SET VARIABLE CHARGED DOLLAR
    SET VARIABLE PAID DOLLAR
    SET VARIABLE TEMPNUM TO VENDNO IN #1
    SET VARIABLE VCURBAL TO CURRBAL IN #1
    COMPUTE PAID AS SUM PD:AMT FROM APTEMP WHERE VENDNO = .TEMPNUM
    IF PAID FAILS THEN
      SET VARIABLE PAID TO 0
    ENDIF
    COMPUTE CHARGED AS SUM T:PRICE FROM APTEMP WHERE VENDNO = .TEMPNUM
    IF CHARGED FAILS THEN
      SET VARIABLE CHARGED TO 0
    ENDIF
    SET VARIABLE VCURBAL TO .VCURBAL + .CHARGED
    SET VARIABLE VCURBAL TO .VCURBAL - .PAID
    CHANGE CURRBAL TO .VCURBAL IN ACCT_PAY WHERE VENDNO = .TEMPNUM
    CHANGE POSTDATE TO .VPOSTDAT IN ACCT_PAY WHERE VENDNO = .TEMPNUM
    NEXT #1 STATUS1
  ENDWHILE  *(-- end of posting individual accounts payable account)
*(-- move journal entries to histical tables)
APPEND AP_CHG TO APCGHIST
APPEND AP_PAID TO APPDHIST
*(--prepare accounts payable journal tables for next month)
DELETE ROWS FROM AP_CHG WHERE VENDNO EXISTS
DELETE ROWS FROM AP_PAID WHERE VENDNO EXISTS
DELETE ROWS FROM APTEMP WHERE LIMIT = 10000

*(* USER STATUS REPORT)
NEWPAGE
WRITE "ACCOUNTS PAYABLE ACCOUNTS POSTED"  AT 10,24
WRITE "WORKING ON BALANCE SHEET ACCOUNTS NOW"  AT 15,21

*(* POSTING BALANCE SHEET ACCOUNTS)
SET VARIABLE VMBRBAL DOLLAR
SET VARIABLE VLESBAL DOLLAR
SET VARIABLE VLESOR TO "1211"
SET VARIABLE VMBRS TO "1210"
COMPUTE VMBRBAL AS SUM CURR_BAL FROM ACCT_REC WHERE MEM_NUM EXISTS
IF VMBRBAL FAILS THEN
   SET VARIABLE VMBRBAL TO 0
ENDIF
CHANGE BALANCE TO .VMBRBAL IN BAL_SHET WHERE ACCT_NUM = .VMBRS
COMPUTE VLESBAL AS SUM CURR_BAL FROM ACCT_REC WHERE LESOR_NU EXISTS
IF VLESBAL FAILS THEN
   SET VARIABLE VLESBAL TO 0
ENDIF
CHANGE BALANCE TO .VLESBAL IN BAL_SHET WHERE ACCT_NUM = .VLESOR
CLEAR VLESOR
CLEAR VMBRBAL
CLEAR VLESBAL
CLEAR VMBRS
*(-- set var for capital asset accounts)
SET VARIABLE VA/C TO "1650"
SET VARIABLE VEQUIP TO "1651"
SET VARIABLE VA/CDEP TO "1750"
SET VARIABLE VEQDEP TO "1751"
SET VARIABLE VASETAMT DOLLAR
SET VARIABLE VDEPAMT DOLLAR
*(* POSTING DEPRECIATION ENTERED BY MANAGER TO BAL_SHET)
COMPUTE VASETAMT AS SUM INIT_CST FROM CAP_ASET WHERE ACCT_NUM = .VA/C
IF VASETAMT FAILS THEN
   SET VARIABLE VASETAMT TO 0
ENDIF
CHANGE BALANCE TO .VASETAMT IN BAL_SHET WHERE ACCT_NUM = .VA/C
CLEAR VASETAMT
COMPUTE VDEPAMT AS SUM ACC_DEP FROM CAP_ASSET WHERE ACCT_NUM = .VA/CDEP
IF VDEPAMT FAILS THEN
   SER VARIABLE VDEPAMT TO 0
ENDIF
CHANGE BALANCE TO .VDEPAMT IN BAL_SHET WHERE ACCT_NUM = .VA/CDEP
```

```
CLEAR VDEPAMT
CLEAR VA/C
CLEAR VA/CDEP
COMPUTE VASETAMT AS SUM INIT_CST FROM CAP_ASET WHERE ACCT_NUM = .VEQUIP
IF VASETAMT FAILS THEN
   SET VARIABLE VASETAMT TO 0
ENDIF
CHANGE BALANCE TO .VASETAMT IN BAL_SHET WHERE ACCT_NUM = .VEQUIP
CLEAR VASETAMT
CLEAR VEQUIP
COMPUTE VDEPAMT AS SUM ACC_DEP FROM CAP_ASET WHERE ACCT_NUM = .VEQDEP
IF VDEPAMT FAILS THEN
   SET VARIABLE VDEPAMT TO 0
ENDIF
CHANGE BALANCE TO .VDEPAMT IN BAL_SHET WHERE ACCT_NUM = .VEQDEP
CLEAR VDEPAMT
CLEAR VEQDEP
CLEAR VAMT
SET VARIABLE VAP_ACCT TO "2100"
COMPUTE VAMT AS SUM CURRBAL FROM ACCT_PAY
IF VAMT FAILS THEN
   SET VARIABLE VAMT TO 0
ENDIF
CHANGE BALANCE TO .VAMT IN BAL_SHET WHERE ACCT_NUM = .VAP_ACCT
CLEAR VAP_ACCT
CLEAR VAMT
*(-- post all plus balance sheet journal entries to proper acct)
SET POINTER #1 STATUS1 FOR BAL_SHET
WHILE STATUS1 = 0 THEN
  CLEAR VMINAMT
  CLEAR VAMT
  CLEAR VCURBAL
  SET VARIABLE TEMPNUM TO ACCT_NUM IN #1
  SET VARIABLE VCURBAL TO BALANCE IN #1
  COMPUTE VAMT AS SUM DOL:AMT FROM BAL_JOUR WHERE ACCT_NUM = .TEMPNUM +
     AND TX:CODE = .VPLUS
  IF VAMT FAILS THEN
     SET VARIABLE VAMT TO 0
  ENDIF
  SET VARIABLE VCURBAL TO .VCURBAL + .VAMT
  CHANGE BALANCE TO .VCURBAL IN BAL_SHET WHERE ACCT_NUM = .TEMPNUM
  COMPUTE VMINAMT AS SUM DOL:AMT FROM BAL_JOUR WHERE ACCT_NUM EQ +
     TEMPNUM AND TX:CODE = .VMINUS
  IF VMINAMT FAILS THEN
     SET VARIABLE VMINAMT TO 0
  ENDIF
  SET VARIABLE VCURBAL TO .VCURBAL - .VMINAMT
  CHANGE BALANCE TO .VCURBAL IN BAL_SHET WHERE ACCT_NUM = .TEMPNUM
  CHANGE BAL:DATE TO .VPOSTDAT IN BAL_SHET WHERE ACCT_NUM = .TEMPNUM
  NEXT #1 STATUS1
ENDWHILE  *(--all balance sheet updated and new posting date)
*(-- place month's journal entries to history table)
APPEND BAL_JOUR TO JOURHIST
*(-- prepare journal table for next month's transactions)
DELETE ROWS FROM BAL_JOUR WHERE ACCT_NUM EXISTS

NEWPAGE
WRITE "BALANCE SHEET ACCOUNTS POSTED"  AT 10,25
WRITE "NOW POSTING INCOME STATEMENT ACCOUNTS"  AT 15,21
WRITE " XXXXX     XXXX      XX    X"  AT 17,25
WRITE " X        X    X     X    X"  AT 18,25
WRITE " XXXX     X    X         X"   AT 19,25
WRITE "     X    X    X        X"  AT 20,25
WRITE "     X    X    X       X   X"  AT 21,25
WRITE " XXXX      XXXX       X   XX"  AT 22,25

*(* POST INCOME STATEMENT ACCOUNTS)
CHANGE BALANCE TO 0 IN EARNINGS WHERE BALANCE FAILS
SET POINTER #1 STATUS1 FOR EARNINGS
```

```
  WHILE STATUS1 = 0 THEN
    CLEAR VMINAMT
    CLEAR VAMT
    CLEAR VCURBAL
    SET VARIABLE VCURBAL TO BALANCE IN #1
    SET VARIABLE TEMPNUM TO ACCT_NUM IN #1
    COMPUTE VAMT AS SUM REV:AMT FROM INCOME WHERE ACCT_NUM = .TEMPNUM +
       AND TX:CODE = .VPLUS
    IF VAMT FAILS THEN
       SET VARIABLE VAMT TO 0
    ENDIF
    SET VARIABLE VCURBAL TO .VCURBAL + .VAMT
    CHANGE BALANCE TO .VCURBAL IN EARNINGS WHERE ACCT_NUM = .TEMPNUM
    COMPUTE VMINAMT AS SUM REV:AMT FROM INCOME WHERE ACCT_NUM EQ +
       TEMPNUM AND TX:CODE = .VMINUS
    IF VMINAMT FAILS THEN
       SET VARIABLE VMINAMT TO 0
    ENDIF
    SET VARIABLE VCURBAL TO .VCURBAL - .VMINAMT
    CHANGE BALANCE TO .VCURBAL IN EARNINGS WHERE ACCT_NUM = .TEMPNUM
    CHANGE BAL:DATE TO .VPOSTDAT IN EARNINGS WHERE ACCT_NUM = .TEMPNUM
    NEXT #1 STATUS1
  ENDWHILE  *(--all income statement updated and new posting date)
*(-- POSTING EXPENSE ACCOUNT UPDATE)

SET POINTER #2 STATUS2 FOR EARNINGS
  WHILE STATUS2 = 0 THEN
    CLEAR VMINAMT
    CLEAR VAMT
    CLEAR VCURBAL
    SET VARIABLE VCURBAL TO BALANCE IN #2
    SET VARIABLE TEMPNUM TO ACCT_NUM IN #2
   COMPUTE VAMT AS SUM EXP:AMT FROM EXPENSE WHERE ACCT_NUM = .TEMPNUM +
       AND TX:CODE = .VPLUS
    IF VAMT FAILS THEN
       SET VARIABLE VAMT TO 0
    ENDIF
    SET VARIABLE VCURBAL TO .VCURBAL + .VAMT
    CHANGE BALANCE TO .VCURBAL IN EARNINGS WHERE ACCT_NUM = .TEMPNUM
    COMPUTE VMINAMT AS SUM EXP:AMT FROM EXPENSE WHERE ACCT_NUM EQ +
       TEMPNUM AND TX:CODE = .VMINUS
    IF VMINAMT FAILS THEN
       SE T VARIABLE VMINAMT TO 0
    ENDIF
    SET VARIABLE VCURBAL TO .VCURBAL - .VMINAMT
    CHANGE BALANCE TO .VCURBAL IN EARNINGS WHERE ACCT_NUM = .TEMPNUM
    CHANGE BAL:DATE TO .VPOSTDAT IN EARNINGS WHERE ACCT_NUM = .TEMPNUM
    NEXT #2 STATUS2
  ENDWHILE  *(--all expenses updated and new posting date)
*(-- transfer of journal entries to history tables)
APPEND INCOME TO INC_HIST
APPEND EXPENSE TO EXP_HIST
*(-- clear journal tables for next month's transactions)
DELETE ROWS FROM INCOME WHERE ACCT_NUM EXISTS
DELETE ROWS FROM EXPENSE WHERE ACCT_NUM EXISTS

NEWPAGE
WRITE "POSTING COMPLETE ON INCOME STATEMENT ACCOUNTS"  AT 10,17
WRITE "NOW WORKING ON FLIGHT HOURS"  AT 15,27
WRITE "XXXXXX    XXXXX     XX   X"  AT 17,25
WRITE "     X    X         X   X"  AT 18,25
WRITE "     X    XXXX        X"   AT 19,25
WRITE "     X        X      X"  AT 20,25
WRITE "     X        X     X   X"  AT 21,25
WRITE "     X    XXXX      X   XX"  AT 22,25

*(* UPDATE FLIGHT HOURS)
*(* A/C_HRS AND A/CMAINT HOURS WERE UPDATE DURING EXECUTION OF)
*(* LES_STMT.CMD AND ACSTAT.CMD MODULES)
*(* A/C_HRS TRANSFERRED TO HOURHIST IN ACSTAT.CMD MODULE )
APPEND A/CMAINT TO MAINHIST
DELETE ROWS FROM A/CMAINT WHERE A/C_NUM EXISTS
APPEND FLT_REC TO FLT_HIST
*(-- compute cumulative flt_hrs by category and faa_cert )
SET VARIABLE VSDCT INTEGER
SET VARIABLE VPRCT INTEGER
SET VARIABLE VCMCT INTEGER
SET VARIABLE VCFCT INTEGER
SET VARIABLE VTOTCT INTEGER
SET POINTER #1 STATUS1 FOR MBR_SUM
  WHILE STATUS1 = 0 THEN
    CLEAR VSDCT
    CLEAR VPRCT
    CLEAR VCMCT
    CLEAR VCFCT
    CLEAR VSHOURS
    CLEAR VPHOURS
    CLEAR VCHOURS
    CLEAR VCFHOURS
    CLEAR VSDHRS
    CLEAR VPRHRS
    CLEAR VCMHRS
    CLEAR VCFHRS
    CLEAR VTOTCT
    CLEAR VTOTNO
    CLEAR VTOTHRS
    CLEAR VTOTFLT
    SET VARIABLE TEMPCAT TO CATEGORY IN #1
    SET VARIABLE VSDHRS TO STUDHRS IN #1
    SET VARIABLE VPRHRS TO PRIVHRS IN #1
    SET VARIABLE VCMHRS TO COMMHRS IN #1
    SET VARIABLE VCFHRS TO CFIHRS IN #1
    SET VARIABLE VTOTCT TO TOTCOUNT IN #1
    SET VARIABLE VTOTHRS TO TOTHOURS IN #1
   COMPUTE VSDCT AS COUNT FAA_CERT FROM MEM_REC WHERE FAA_CERT = STUD +
      AND CATEGORY = .TEMPCAT
   IF VSDCT FAILS THEN
      SET VARIABLE VSDCT TO 0
   ENDIF
   COMPUTE VPRCT AS COUNT FAA_CERT FROM MEM_REC WHERE FAA_CERT = PRIV +
      AND CATEGORY = .TEMPCAT
   IF VPRCT FAILS THEN
      SET VARIABLE VPRCT TO 0
   ENDIF
   COMPUTE VCMCT AS COUNT FAA_CERT FROM MEM_REC WHERE FAA_CERT = COMM +
      AND CATEGORY = .TEMPCAT
   IF VCMCT FAILS THEN
      SET VARIABLE VCMCT TO 0
   ENDIF
   COMPUTE VCFCT AS COUNT FAA_CERT FROM MEM_REC WHERE FAA_CERT = CFI +
      AND CATEGORY = .TEMPCAT
   IF VCFCT FAILS THEN
      SET VARIABLE VCFCT TO 0
   ENDIF

   COMPUTE VSHOURS AS SUM FLT_HRS FROM FLT_REC WHERE FAA_CERT = STUD +
      AND CATEGORY = .TEMPCAT
   IF VSHOURS FAILS THEN
      SET VARIABLE VSHOURS TO 0
   ENDIF
   COMPUTE VPHOURS AS SUM FLT_HRS FROM FLT_REC WHERE FAA_CERT = PRIV +
      AND CATEGORY = .TEMPCAT
   IF VPHOURS FAILS THEN
      SET VARIABLE VPHOURS TO 0
   ENDIF
   COMPUTE VCHOURS AS SUM FLT_HRS FROM FLT_REC WHERE FAA_CERT = COMM +
      AND CATEGORY = .TEMPCAT
   IF VCHOURS FAILS THEN
```

```
      SET VARIABLE VCHOURS TO 0
   ENDIF
   COMPUTE VCFHOUR AS SUM FLT_HRS FROM FLT_REC WHERE FAA_CERT = CFI +
      AND CATEGORY = .TEMPCAT
   IF VCFHOUR FAILS THEN
      SET VARIABLE VCFHOUR TO 0
   ENDIF
   SET VARIABLE VSDHRS TO .VSDHRS + .VSHOURS
   SET VARIABLE VPRHRS TO .VPRHRS + .VPHOURS
   SET VARIABLE VCMHRS TO .VCMHRS + .VCHOURS
   SET VARIABLE VCFHRS TO .VCFHRS + .VCFHOUR
   SET VARIABLE NUM1 TO .VSDCT
   SET VARIABLE NUM2 TO .VPRCT
   SET VARIABLE NUM3 TO .NUM1 + .NUM2
   SET VARIABLE NUM4 TO .NUM3 + .VCMCT
   SET VARIABLE VTOTNO TO .NUM4 + .VCFCT
   SET VARIABLE VTOTCT TO .VTOTCT + .VTOTNO
   SET VARIABLE HRS1 TO .VSDHRS
   SET VARIABLE HRS2 TO .VPRHRS
   SET VARIABLE HRS3 TO .HRS1 + .HRS2
   SET VARIABLE HRS4 TO .HRS3 + .VCMHRS
   SET VARIABLE VTOTFLT TO .HRS4 + .VCFHRS
   SET VARIABLE VTOTHRS TO .VTOTHRS + .VTOTFLT
   CHANGE SUMDATE TO .VPOSTDAT IN MBR_SUM WHERE CATEGORY = .TEMPCAT
   CHANGE STUDCT TO .VSDCT IN MBR_SUM WHERE CATEGORY = .TEMPCAT
   CHANGE STUDHRS TO .VSDHRS IN MBR_SUM WHERE CATEGORY = .TEMPCAT
   CHANGE PRIVCT TO .VPRCT IN MBR_SUM WHERE CATEGORY = .TEMPCAT
   CHANGE PRIVHRS TO .VPRHRS IN MBR_SUM WHERE CATEGORY = .TEMPCAT
   CHANGE COMMCT TO .VCMCT IN MBR_SUM WHERE CATEGORY = .TEMPCAT
   CHANGE COMMHRS TO .VCMHRS IN MBR_SUM WHERE CATEGORY = .TEMPCAT
   CHANGE CFICT TO .VCFCT IN MBR_SUM WHERE CATEGORY = .TEMPCAT
   CHANGE CFIHRS TO .VCFHRS IN MBR_SUM WHERE CATEGORY = .TEMPCAT
   CHANGE TOTCOUNT TO .VTOTCT IN MBR_SUM WHERE CATEGORY = .TEMPCAT
   CHANGE TOTHOURS TO .VTOTHRS IN MBR_SUM WHERE CATEGORY = .TEMPCAT
   NEXT #1 STATUS1
ENDWHILE *(* posting cumulative flight hours by category and faa_cert)

*(--prepare table for next month transactions)
DELETE ROWS FROM FLT_REC WHERE LIMIT = 10000

NEWPAGE
WRITE "POSTING COMPLETED"  AT 10,30
WRITE "TO CLEAN UP THE DATABASE FROM ALL THE DELETIONS AND"  AT 15,14
WRITE "MOVES, INSERT THE FIRST FORMATED DISK LABELED FOR THE" AT 16,13
WRITE "NEXT MONTH TO MAKE A BACKUP COPY OF THE DATABASE AND"  AT 17,12
WRITE "REPACK THE DISK TO INCREASE EFFICIENCY."  AT 18,14
WRITE "PRESS ANY KEY ONCE YOU HAVE A FORMATED DISK IN THE"  AT 20,14
WRITE "'A' DISK DRIVE AND WE WILL CONTINUE."  AT 21,22
PAUSE
*(-- set messages on inorder to tell if error in copying database)
*(-- such as full disk, etc)
SET ERROR MESSAGES ON
SET ERR VAR ERV
COPY FLYCLUB1.RBS A:
COPY FLYCLUB3.RBS A:
NEWPAGE
WRITE "PLACE NEXT FORMATED DISK IN DRIVE TO COMPLETE COPY" AT 10,10
WRITE "PRESS ANY KEY TO CONTINUE"  AT 15,25
PAUSE
COPY FLYCLUB2.RBS A:
IF ERV = 0 THEN
    WRITE "PACKING DATABASE AT THIS TIME"  AT 20,25
    PACK FLYCLUB
    IF ERV = 0 THEN
        NEWPAGE
        WRITE "PLACE FIRST DATABASE DISK IN DRIVE NOW"  AT 10,20
        WRITE "PRESS ANY KEY TO CONTINUE"  AT 15,25
        PAUSE
        COPY FLYCLUB1.RBS A:
```

```
        COPY FLYCLUB3.RBS A:
        WRITE "PLACE SECOND DATABASE DISK IN DRIVE NOW"  AT 20,20
        WRITE "PRESS ANY KEY TO CONTINUE"  AT 22,25
        PAUSE
        COPY FLYCLUB2.RBS A:
    ENDIF
ELSE
    NEWPAGE
    WRITE "TRY ANOTHER DISK, UNABLE TO BACKUP DATABASE" AT 15,10
    WRITE "PRESS ANY KEY TO CONTINUE" AT 17,14
    PAUSE
ENDIF

NEWPAGE
WRITE "THE PACK IS COMPLETE, AT 5,25
WRITE "THE DISKS COPIED NOW CONTAIN A BACKUP OF THE DATABASE" AT 7,10
WRITE "PRESS ANY KEY TO RETURN TO MENU" AT 9,24
PAUSE

LABEL MONTHEND
SET MESSAGES ON
SET ERROR MESSAGES ON
RETURN
*(* END OF END_MON.CMD )
$COMMAND
BACKPACK
*(*********************************************************************
PROGRAM:      BACKPACK.CMD
AUTHOR:       J. M. GRAHAM
DATE:         NOV 1986      VER. 1.8
DESCRIPTION:  CREATES A BACKUP OF THE DATABASE AND REPACKS IT
                                                                   (11)
**********************************************************************)

SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

NEWPAGE
WRITE "INSERT A FORMATTED DISK IN DRIVE A" AT 5,10
WRITE "PRESS ANY KEY TO CONTINUE" AT 7,14
PAUSE
*(-- set messages on inorder to tell if error in copying database)
*(-- such as full disk, etc)
SET ERROR MESSAGES ON
SET ERR VAR ERV
COPY FLYCLUB1.RBS A:
COPY FLYCLUB3.RBS A:
NEWPAGE
WRITE "PLACE NEXT FORMATED DISK IN DRIVE TO COMPLETE COPY" AT 10,10
WRITE "PRESS ANY KEY TO CONTINUE"  AT 15,25
PAUSE
COPY FLYCLUB2.RBS A:
IF ERV = 0 THEN
    WRITE "PACKING DATABASE AT THIS TIME"  AT 20,25
    PACK FLYCLUB
    IF ERV = 0 THEN
        NEWPAGE
        WRITE "PLACE FIRST DATABASE DISK IN DRIVE NOW"  AT 10,20
        WRITE "PRESS ANY KEY TO CONTINUE"  AT 15,25
        PAUSE
        COPY FLYCLUB1.RBS A:
        COPY FLYCLUB3.RBS A:
        WRITE "PLACE SECOND DATABASE DISK IN DRIVE NOW"  AT 20,20
        WRITE "PRESS ANY KEY TO CONTINUE"  AT 22,25
        PAUSE
        COPY FLYCLUB2.RBS A:
    ENDIF
ELSE
```

```
    NEWPAGE
    WRITE "TRY ANOTHER DISK, UNABLE TO BACKUP DATABASE" AT 15,10
    WRITE "PRESS ANY KEY TO CONTINUE" AT 17,14
    PAUSE
ENDIF

NEWPAGE
WRITE "THE PACK IS COMPLETE, AT 5,25
WRITE "THE DISKS COPIED NOW CONTAIN A BACKUP OF THE DATABASE" AT 7,10
WRITE "PRESS ANY KEY TO CONTINUE" AT 9,20
PAUSE

LABEL BACKEND
SET MESSAGES ON
RETURN
*(* END BACKPACK.CMD )

$COMMAND
MEMSTMT
*(*****************************************************************
PROGRAM:       MEMSTMT.CMD
AUTHOR:        J. M. GRAHAM
DATE:          JAN 1987       VER 1.2
DESCRIPTION:   THIS MODULE COMPUTES THE ACTIVE MEMBERS' STATEMENT
               CHARGES, PRINTS THE MEMBERS' STATEMENTS, UPDATES THE
               CLUB'S ACCOUNT RECEIVABLE RECORDS, AND POST ALL TABLES
               AND PREPARES FOR THE NEW MONTHLY TRANSACTIONS.

TABLES USED:   MEM_REC, MEM_CHG, MEM_PAY, CHGHIST, PAYHIST,
               BILLTEMP, ACCT_REC
FORMS USED:    NONE
REPORTS USED:  MEM_BILL                                         (14)
******************************************************************)

NEWPAGE
WRITE "STATEMENTS ARE TO BE PRINTED ONCE A MONTH ONLY." AT 5,12
WRITE "REOCCURRING CHARGES MUST BE DONE BEFORE BILLING." AT 7,10
WRITE "ENSURE A BACKUP COPY OF THE DATABASE HAS BEEN MADE." AT 9,10
WRITE "ENSURE YOUR PRINTER IS READY."  AT 11,15
WRITE "DO YOU WANT TO PROCEED? (Y/N) - " AT 13,17
FILLIN YOURANS USING " " AT 13,50
IF YOURANS = N THEN
   GOTO BAILOUT
ENDIF

NEWPAGE
WRITE "PREPARING MEMBERS' STATEMENTS"  AT 5,25
WRITE "-----------------------------"  AT 6,25
WRITE "ENTER:"  AT 8,10
WRITE "1 - PREPARE/PRINT ALL MEMBERS' STATEMENTS"  AT 10,10
WRITE "2 - PREPARE/PRINT ONE MEMBER STATEMENT**SPECIAL CASE**" AT 12,10
WRITE "3 - EXIT ... RETURN TO END-OF-THE-MONTH MENU"  AT 14,10
WRITE "CHOICE:"  AT 16,10
FILLIN YOURCH USING " " AT 16,20
IF YOURCH = 1 THEN
   GOTO NORMBILL
ENDIF
IF YOURCH = 2 THEN
   GOTO ONEBILL
ENDIF
IF YOURCH = 3 THEN
   GOTO BAILOUT
ENDIF

LABEL NORMBILL
NEWPAGE
WRITE "ENSURE PRINTER IS READY."  AT 10,20
WRITE "WORKING ON MEMBERS' STATEMENTS"  AT 15,15
SET MESSAGES OFF
```

```
OPEN FLYCLUB
SET ERROR MESSAGES OFF

*(Perform account updates in preparation for this month's billing )
*(First, shift back Current, 30, 60, and 90 day balances)
ASSIGN BAL:90 TO BAL:90 + BAL:60 IN ACCT_REC
ASSIGN BAL:60 TO BAL:30 + $0.00 IN ACCT_REC
ASSIGN SUM60+90 TO BAL:60 + BAL:90 IN ACCT_REC
ASSIGN BAL:30 TO CURR_BAL - SUM60+90 IN ACCT_REC
ASSIGN BAL_FWD TO CURR_BAL + $0.00 IN ACCT_REC
ASSIGN CURR_BAL TO 0 IN ACCT_REC WHERE MEM_NUM EXISTS
ASSIGN BALFWD30 TO BAL:30 + $0.00 IN ACCT_REC
ASSIGN BALFWD60 TO BAL:60 + $0.00 IN ACCT_REC
ASSIGN BALFWD90 TO BAL:90 + $0.00 IN ACCT_REC

*(-- Establish BILLTEMP table)
DELETE ROWS FROM BILLTEMP WHERE LIMIT = 10000
APPEND MEM_CHG TO BILLTEMP
APPEND MEM_PAY TO BILLTEMP

*(-- Establish loop to process payments received this billing period.)
SET POINTER #1 STATUS1 FOR MEM_REC
WHILE STATUS1 = 0 THEN
    SET VARIABLE CHARGED DOLLAR
    SET VARIABLE PAID DOLLAR
    SET VARIABLE TEMPNUM1 TO MEM_NUM IN #1
   *(-- replace all nulls with 0's )
   CHANGE BAL:90 TO 0 IN ACCT_REC WHERE BAL:90 FAILS
   CHANGE BAL:60 TO 0 IN ACCT_REC WHERE BAL:60 FAILS
   CHANGE BAL:30 TO 0 IN ACCT_REC WHERE BAL:30 FAILS
   CHANGE CURR_BAL TO 0 IN ACCT_REC WHERE CURR_BAL FAILS
   CHANGE BALFWD30 TO 0 IN ACCT_REC WHERE BALFWD30 FAILS
   CHANGE BALFWD60 TO 0 IN ACCT_REC WHERE BALFWD60 FAILS
   CHANGE BALFWD90 TO 0 IN ACCT_REC WHERE BALFWD90 FAILS

   *(-- Establish variables to work with)
   SET VARIABLE VBAL:90 TO BAL:90 IN ACCT_REC WHERE MEM_NUM = .TEMPNUM1
   SET VARIABLE VBAL:60 TO BAL:60 IN ACCT_REC WHERE MEM_NUM = .TEMPNUM1
   SET VARIABLE VBAL:30 TO BAL:30 IN ACCT_REC WHERE MEM_NUM = .TEMPNUM1
 SET VARIABLE VCUR_BAL TO CURR_BAL IN ACCT_REC WHERE MEM_NUM = .TEMPNUM1
   COMPUTE PAID AS SUM PAYMENTS FROM BILLTEMP WHERE MEM_NUM = .TEMPNUM1
   IF PAID FAILS THEN
       SET VAR PAID TO $0.00
    ENDIF
COMPUTE CHARGED AS SUM CHARGES FROM BILLTEMP WHERE MEM_NUM = .TEMPNUM1

  *(-- Apply payments to past due balances )
    IF PAID >= .VBAL:90 THEN
       SET VAR PAID TO .PAID - .VBAL:90
       SET VAR VBAL:90 TO 0
          IF PAID >= .VBAL:60 THEN
             SET VAR PAID TO .PAID - .VBAL:60
             SET VAR VBAL:60 TO 0
           IF PAID >= .VBAL:30 THEN
              SET VAR PAID TO .PAID - .VBAL:30
              SET VAR VBAL:30 TO $0.00
              SET VAR VCUR_BAL TO .VBAL:30 + .CHARGED - .PAID
           ELSE  *(-- Past due 30 balance > payment )
              SET VAR VBAL:30 TO .VBAL:30 - .PAID
              SET VAR PAID TO 0
              SET VAR VCUR_BAL TO .VBAL:60 + .VBAL:30
              SET VAR VCUR_BAL TO .VCUR_BAL + .CHARGED
           ENDIF  *(-- Past due 30 balance > payment )
         ELSE  *(-- Past due 60 balance > payment )
            SET VAR VBAL:60 TO .VBAL:60 - .PAID
            SET VAR PAID TO 0
            SET VAR VCUR_BAL TO .VBAL:60 + .VBAL:30
            SET VAR VCUR_BAL TO .VCUR_BAL + .CHARGED
         ENDIF  *(-- Past due 60 balance > payment )
    ELSE   *(-- Past due 90 balance > payment )
       SET VAR VBAL:90 TO .VBAL:90 - .PAID
       SET VAR PAID TO 0
       SET VAR VCUR_BAL TO .VBAL:90 + .VBAL:60
       SET VAR VCUR_BAL TO .VCUR_BAL + .VBAL:30
       SET VAR VCUR_BAL TO .VCUR_BAL + .CHARGED
    ENDIF  *(-- past due 90 balance > payment )

  *(-- Enter new values in member's record )
    CHANGE BAL:90 TO .VBAL:90 IN ACCT_REC WHERE MEM_NUM = .TEMPNUM1
    CHANGE BAL:60 TO .VBAL:60 IN ACCT_REC WHERE MEM_NUM = .TEMPNUM1
    CHANGE BAL:30 TO .VBAL:30 IN ACCT_REC WHERE MEM_NUM = .TEMPNUM1
    CHANGE CURR_BAL TO .VCUR_BAL IN ACCT_REC WHERE MEM_NUM = .TEMPNUM1
    NEXT #1 STATUS1
ENDWHILE  *(-- loop to update payments this period )

*(-- Print member statements)
OUTPUT PRINTER
SET NULL " "
SET POINTER #2 STATUS2 FOR MEM_REC
   WHILE STATUS2 = 0 THEN
     SET VARIABLE TEMPNUM2 TO MEM_NUM IN #2
     PRINT MEM_BILL SORTED BY L:NAME F:NAME WHERE MEM_NUM EQ .TEMPNUM2
     NEXT #2 STATUS2
   ENDWHILE  *(-- Print member statements )
OUTPUT SCREEN
SET NULL -0-

*(-- Move transactions to history tables)
APPEND MEM_CHG TO CHGHIST
APPEND MEM_PAY TO PAYHIST

*(-- Prepare tables for next month's entries)
DELETE ROWS FROM MEM_CHG WHERE MEM_NUM EXISTS
DELETE ROWS FROM MEM_PAY WHERE MEM_NUM EXISTS
*(-- DELETE ROWS FROM MEM_CHG WHERE DESCRIPT = DUES )

*(-- Updates complete)

NEWPAGE
WRITE "Monthly Billing and Accounts Update Complete." AT 7,10
WRITE "PRESS ANY KEY TO RETURN TO MENU." AT 10,15
PAUSE
GOTO BAILOUT
*(-- end of normal billing subroutine )

LABEL ONEBILL
NEWPAGE
SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF
WRITE "PREPARING AND PRINTING ONE MEMBER'S STATEMENT"  AT 3,15
WRITE "---------------------------------------------"  AT 4,15
WRITE "%%%% SPECIAL CASE BILLING !! %%%%"  AT 7,20
WRITE "IT IS ASSUMED THAT THE MEMBER IS LEAVING IN MID MONTH"  AT 9,5
WRITE "HIS/HER MEMBER STATUS WILL BE CHANGED TO NONACTIVE"  AT 10,5
WRITE "THIS WAY HE/SHE WILL NOT BE DOUBLED CHARGED FOR DUES"  AT 11,5
WRITE "THEREFORE, YOU WILL NEED TO PROVIDE SOME DATA: "  AT 13,10
LABEL GETDATA
FILLIN VMEMNUM USING "BILLING MEMBER NUMBER - "  AT 15,20
FILLIN VL:NAME USING "MEMBER'S LAST NAME IS - "  AT 17,20
SET VARIABLE VDUECHG DOLLAR
FILLIN VDUECHG USING "AMOUNT OF DUES TO BE CHARGED IS - $"  AT 19,20
SET POINTER #1 STATUS1 FOR MEM_REC WHERE MEM_NUM EQ .VMEMNUM AND +
    L:NAME EQ .VL:NAME
IF STATUS1 <> 0 THEN
  WRITE "NO SUCH MEMBER'S ACCOUNT"  AT 21,20
  WRITE "PRESS ANY KEY TO REENTER DATA"  AT 22,15
  PAUSE
  NEWPAGE
  CLEAR VMEMNUM
  CLEAR VL:NAME
  CLEAR VDUECHG
  GOTO GETDATA
ENDIF

NEWPAGE
WRITE "ENSURE PRINTER IS READY."  AT 10,20
WRITE "WORKING ON MEMBERS' STATEMENTS"  AT 15,15

*(Perform account updates in preparation for this month's billing )
*(First, shift back Current, 30, 60, and 90 day balances)
ASSIGN BAL:90 TO BAL:90 + BAL:60 IN ACCT_REC WHERE MEM_NUM EQ .VMEMNUM
ASSIGN BAL:60 TO BAL:30 + $0.00 IN ACCT_REC WHERE MEM_NUM EQ .VMEMNUM
ASSIGN SUM60+90 TO BAL:60 + BAL:90 IN ACCT_REC WHERE MEM_NUM = .VMEMNUM
ASSIGN BAL:30 TO CURR_BAL-SUM60+90 IN ACCT_REC WHERE MEM_NUM = .VMEMNUM
ASSIGN BAL_FWD TO CURR_BAL + $0.00 IN ACCT_REC WHERE MEM_NUM = .VMEMNUM
ASSIGN CURR_BAL TO 0 IN ACCT_REC WHERE MEM_NUM WHERE MEM_NUM = .VMEMNUM
ASSIGN BALFWD30 TO BAL:30 + $0.00 IN ACCT_REC WHERE MEM_NUM = .VMEMNUM
ASSIGN BALFWD60 TO BAL:60 + $0.00 IN ACCT_REC WHERE MEM_NUM = .VMEMNUM
ASSIGN BALFWD90 TO BAL:90 + $0.00 IN ACCT_REC WHERE MEM_NUM = .VMEMNUM

*(-- charging dues to member leaving in mid month)
SET VARIABLE VACCTNUM TO "8100"
SET VARIABLE VCHARGES DOLLAR
SET VARIABLE VTRANDAT TO .#DATE
SET VARIABLE VCHARGES TO .VDUECHG
LOAD MEM_CHG USING MEM_NUM TRANDATE CHARGES DESCRIPT
   .VMEMNUM .VTRANDAT .VCHARGES DUES
END
*(-- post income from dues of member leaving midmonth)
LOAD INCOME
   .VACCTNUM DUES .VCHARGES .VTX:DATE P
END

*(-- Establish BILLTEMP table)
DELETE ROWS FROM BILLTEMP WHERE LIMIT = 10000
APPEND MEM_CHG TO BILLTEMP WHERE MEM_NUM EQ .VMEMNUM
APPEND MEM_PAY TO BILLTEMP WHERE MEM_NUM EQ .VMEMNUM

*(-- Establish loop to process payments received this billing period.)
SET POINTER #2 STATUS2 FOR MEM_REC WHERE MEM_NUM EQ .VMEMNUM
WHILE STATUS2 = 0 THEN
    SET VARIABLE CHARGED DOLLAR
    SET VARIABLE PAID DOLLAR
    SET VARIABLE TEMPNUM2 TO MEM_NUM IN #2
   *(-- replace all nulls with 0's )
   CHANGE BAL:90 TO 0 IN ACCT_REC WHERE BAL:90 FAILS
   CHANGE BAL:60 TO 0 IN ACCT_REC WHERE BAL:60 FAILS
   CHANGE BAL:30 TO 0 IN ACCT_REC WHERE BAL:30 FAILS
   CHANGE CURR_BAL TO 0 IN ACCT_REC WHERE CURR_BAL FAILS
   CHANGE BALFWD30 TO 0 IN ACCT_REC WHERE BALFWD30 FAILS
   CHANGE BALFWD60 TO 0 IN ACCT_REC WHERE BALFWD60 FAILS
   CHANGE BALFWD90 TO 0 IN ACCT_REC WHERE BALFWD90 FAILS

   *(-- Establish variables to work with)
   SET VARIABLE VBAL:90 TO BAL:90 IN ACCT_REC WHERE MEM_NUM = .TEMPNUM2
   SET VARIABLE VBAL:60 TO BAL:60 IN ACCT_REC WHERE MEM_NUM = .TEMPNUM2
   SET VARIABLE VBAL:30 TO BAL:30 IN ACCT_REC WHERE MEM_NUM = .TEMPNUM2
  SET VARIABLE VCUR_BAL TO CURR_BAL IN ACCT_REC WHERE MEM_NUM = .TEMPNUM2
   COMPUTE PAID AS SUM PAYMENTS FROM BILLTEMP WHERE MEM_NUM = .TEMPNUM2
   IF PAID FAILS THEN
       SET VAR PAID TO $0.00
   ENDIF
COMPUTE CHARGED AS SUM CHARGES FROM BILLTEMP WHERE MEM_NUM = .TEMPNUM2

  *(-- Apply payments to past due balances )
    IF PAID >= .VBAL:90 THEN
```

```
      SET VAR PAID TO .PAID - .VBAL:90
      SET VAR VBAL:90 TO 0
         IF PAID >= .VBAL:60 THEN
            SET VAR PAID TO .PAID - .VBAL:60
            SET VAR VBAL:60 TO 0
           IF PAID >= .VBAL:30 THEN
              SET VAR PAID TO .PAID - .VBAL:30
              SET VAR VBAL:30 TO $0.00
              SET VAR VCUR_BAL TO .VBAL:30 + .CHARGED - .PAID
           ELSE  *(-- Past due 30 balance > payment )
              SET VAR VBAL:30 TO .VBAL:30 - .PAID
              SET VAR PAID TO 0
              SET VAR VCUR_BAL TO .VBAL:60 + .VBAL:30
              SET VAR VCUR_BAL TO .VCUR_BAL + .CHARGED
           ENDIF  *(-- Past due 30 balance > payment )
         ELSE  *(-- Past due 60 balance > payment )
            SET VAR VBAL:60 TO .VBAL:60 - .PAID
            SET VAR PAID TO 0
            SET VAR VCUR_BAL TO .VBAL:60 + .VBAL:30
            SET VAR VCUR_BAL TO .VCUR_BAL + .CHARGED
         ENDIF  *(-- Past due 60 balance > payment )
   ELSE   *(-- Past due 90 balance > payment )
      SET VAR VBAL:90 TO .VBAL:90 - .PAID
      SET VAR PAID TO 0
      SET VAR VCUR_BAL TO .VBAL:90 + .VBAL:60
      SET VAR VCUR_BAL TO .VCUR_BAL + .VBAL:30
      SET VAR VCUR_BAL TO .VCUR_BAL + .CHARGED
   ENDIF  *(-- past due 90 balance > payment )

  *(-- Enter new values in member's record )
   CHANGE BAL:90 TO .VBAL:90 IN ACCT_REC WHERE MEM_NUM = .TEMPNUM2
   CHANGE BAL:60 TO .VBAL:60 IN ACCT_REC WHERE MEM_NUM = .TEMPNUM2
  CHANGE BAL:30 TO .VBAL:30 IN ACCT_REC WHERE MEM_NUM = .TEMPNUM2
  CHANGE CURR_BAL TO .VCUR_BAL IN ACCT_REC WHERE MEM_NUM = .TEMPNUM2
  NEXT #2 STATUS2
ENDWHILE  *(-- loop to update payments this period )

*(-- Print member statements)
OUTPUT PRINTER
SET NULL " "
  PRINT MEM_BILL WHERE MEM_NUM EQ .TEMPNUM2
OUTPUT SCREEN
*(-- end of printing member's statement)
SET NULL -0-

*(-- Move transactions to history tables)
APPEND MEM_CHG TO CHGHIST WHERE MEM_NUM EQ .VMEMNUM
APPEND MEM_PAY TO PAYHIST WHERE MEM_NUM EQ .VMEMNUM

*(-- Remove member's data from tables so not to double billing)
SET VARIABLE .VNONACT TO "N" *(-- now inactive membership)
CHANGE MEM_STAT TO .VNONACT IN MEM_REC WHERE MEM_NUM EQ .VMEMNUM
DELETE ROWS FROM MEM_CHG WHERE MEM_NUM EQ .VMEMNUM
DELETE ROWS FROM MEM_PAY WHERE MEM_NUM EQ .VMEMNUM
*(-- DELETE ROWS FROM MEM_CHG WHERE DESCRIPT = DUES )

NEWPAGE
WRITE "SPEICAL BILLING AND ACCOUNT UPDATE COMPLETED"  AT 10,10
WRITE "PRESS ANY KEY TO RETURN TO MENU"  AT 15,15
PAUSE
*(-- Updates complete)
LABEL BAILOUT
SET MESSAGES ON
SET ERROR MESSAGES ON
RETURN
*(* END OF MEMSTMT.CMD )
$COMMAND
INSTSTMT
*(*******************************************************************
PROGRAM:       INSTSTMT.CMD
AUTHOR:        J. M. GRAHAM
DATE:          JAN 1987
DESCRIPTION:   COMPUTES AND PRINTS INSTRUCTOR STATEMENTS AND UPDATES
               THE INSTRUCTOR TABLES FOR THE END OF THE MONTH.

TABLES USED:   INST_REC, INSTHIST
FORMS USED:    NONE
REPORTS USED:  INSTSTMT                                          (11)
*****************************************************************)

NEWPAGE
WRITE "STATEMENTS ARE TO BE PRINTED ONCE A MONTH ONLY." AT 5,12
WRITE "MONTHLY UPDATES ARE AUTOMATICALLY PERFORMED." AT 7,13
WRITE "ENSURE A BACKUP COPY OF THE DATABASE HAS BEEN MADE." AT 9,10
WRITE "DO YOU WANT TO PROCEED? (Y/N) - " AT 11,17
FILLIN YOURANS USING " " AT 11,50
IF YOURANS = N THEN
   GOTO BAILOUT
ENDIF

NEWPAGE
WRITE "ENSURE PRINTER IS READY."  AT 10,20
WRITE "PRESS ANY KEY TO CONTINUE." AT 15,20
PAUSE

NEWPAGE
WRITE "WORKING ON INSTRUCTOR STATEMENTS"  AT 10,15
SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

*(-- Print instructor statements )
OUTPUT PRINTER
SET NULL " "
SET POINTER #1 STATUS1 FOR MEM_REC SORTED BY L:NAME F:NAME WHERE +
    INST_NUM EXISTS
    WHILE STATUS1 = 0 THEN
         SET VARIABLE TEMPNUM1 TO INST_NUM IN #1
         PRINT INSTSTMT WHERE INST_NUM = .TEMPNUM1
         NEXT #1 STATUS1
    ENDWHILE  *(-- Print instructor statements)
OUTPUT SCREEN
SET NULL -0-

*(-- Move transactions to history table)
APPEND INST_REC TO INSTHIST

*(-- Prepare table for next month's entries)
DELETE ROWS FROM INST_REC WHERE INST_NUM EXISTS
*(-- Updates complete)

NEWPAGE
WRITE "Monthly Instructor Statements Complete." AT 7,10
WRITE "PRESS ANY KEY TO RETURN TO MENU." AT 10,15
PAUSE

LABEL BAILOUT
  SET MESSAGES ON
  SET ERROR MESSAGES ON
  RETURN
*(*  END OF INSTSTMT.CMD  )
$COMMAND
LES_STMT
*(*******************************************************************
PROGRAM:       LES_STMT.CMD
AUTHOR:        J. M. GRAHAM
DATE:          JAN 1987      VER 1.2
DESCRIPTION:   THIS MODULE COMPUTES AND PRINTS THE LESSOR'S STATEMENTS.
```

```
              THE A/C_HRS AND A/CMAINT TABLES MUST HAVE BEEN KEPT UP
              TO DATE THROUGH THE MONTH.  THE MANAGER SHOULD PREPARE
              THE A/C STATUS REPORT AFTER PRINTING THE LESSOR'S
              STATEMENTS TO ENSURE PROPER UPDATE OF A/C HOURS.

TABLES USED:  LES_ACCT, A/C_REC, A/C_HRS, A/CMAINT
FORMS USED:   NONE                                             (15)
*******************************************************************)

SET MESSAGES OFF
OPEN FLYCLUB
SET ERROR MESSAGES OFF

NEWPAGE
WRITE "STATEMENTS ARE TO BE PRINTED ONCE A MONTH ONLY." AT 5,12
WRITE "ENSURE THE AIRCRAFT HOURS AND MAINTENANCE RECORDS ARE" AT 7,8
WRITE "UPDATED PRIOR TO PRINTING THE LESSOR'S STATEMENTS."  AT 9,10
WRITE "ENSURE A BACKUP COPY OF THE DATABASE HAS BEEN MADE." AT 11,10
WRITE "DO YOU WANT TO PROCEED? (Y/N) - " AT 13,17
FILLIN YOURANS USING " " AT 13,50
IF YOURANS = N THEN
   GOTO BAILOUT
ENDIF

NEWPAGE
WRITE "ENSURE PRINTER IS READY." AT 10,25
WRITE "PRESS ANY KEY TO BEGIN LESSOR STATEMENTS.  AT 15,15
PAUSE

NEWPAGE
WRITE "WORKING ON LESSOR STATEMENTS."  AT 10,20
WRITE "WHY NOT TAKE A COFFEE BREAK." AT 12,20

*(-- Print lessor statements and compute lease payment )
OUTPUT PRINTER
SET NULL " "
SET POINTER #1 STATUS1 FOR A/C_REC WHERE LEAS_RAT EXISTS
   WHILE STATUS1 = 0 THEN
   SET VARIABLE TEMPNUM1 TO A/C_NUM IN #1
  COMPUTE LABOR AS SUM LABR_CHG FROM A/CMAINT WHERE A/C_NUM = .TEMPNUM1
  COMPUTE PART AS SUM PART_CHG FROM A/CMAINT WHERE A/C_NUM = .TEMPNUM1
  SET VARIABLE ALLMAINT TO .LABOR + .PART
  CHANGE TOTMAINT TO .ALLMAINT IN A/C_HRS WHERE A/C_NUM = .TEMPNUM1
  PRINT LES_STMT WHERE A/C_NUM = .TEMPNUM1
  PRINT MAINT WHERE A/C_NUM = .TEMPNUM1
  *(-- Determine lease payment due)
  SET VARIABLE VTIME TO HOBLEASE IN A/C_HRS WHERE A/C_NUM = .TEMPNUM1
  SET VARIABLE VRATE DOLLAR
  SET VARIABLE VRATE TO LEAS_RAT IN #1
  SET VARIABLE VAMOUNT DOLLAR
  SET VARIABLE VAMOUNT TO .VTIME X .VRATE
  CHANGE LEASEAMT TO .VAMOUNT IN A/C_HRS WHERE A/C_NUM = .TEMPNUM1
  *(-- Determine lease payment due completed)
   NEXT #1 STATUS1
 ENDWHILE  *(-- Print lessor statements)
OUTPUT SCREEN
SET NULL -0-

NEWPAGE
WRITE "LESSOR STATEMENTS COMPLETED."  AT 7,20
WRITE "PRESS ANY KEY TO RETURN TO MENU"  AT 10,15
PAUSE

LABEL Bailout
SET MESSAGES ON
SET ERROR MESSAGES ON
RETURN
*(*  END OF LES_STMT.CMD )
```

LIST OF REFERENCES

1. George, Derek R., *The Design and Development of a Management Information System for the Monterey Navy Flying Club*, Masters Thesis, Naval Postgradute School, Monterey, California, March 1986.

2. Chief of Naval Operations, OPNAV Instruction 1710.2C, "Navy Flying Club Program," October 9, 1984.

3. Weinberg, Victor, *Structured Analysis*, Yourdon Press, 1978.

4. Simpson, Alan, *Understanding R:BASE 5000*, SYBEX, Berkeley, California, 1985.

5. Baran, Nicholas M., "Express Yourself," *PC World*, pp. 229-235, January 1986.

6. Urschel, William, "R:Base: The Promise Expressed," *PC World*, pp. 230-239, February 1987.

7. Page-Jones, Meilir, *The Practical Guide to Structural Systems Design*, pp. 57-70, Yourdon Press, 1980.

BIBLIOGRAPHY

Keen, Peter G. W., and Morton, Michael S. Scott, *Decision Support Systems: An Organization Perspective*, Reading, Mass.: Addison-Wesley Publishing Co. INC., 1978.

Sprague, Ralph H. Jr., and Carlson, Eric D., *Building Effective Decision Support Systems*, Englewood Cliffs, NJ.: Prentice-Hall, INC., 1982.

DeMarco, Tom, *Controlling Software Projects*, New York: Yourdon Press, 1982.

Weinberg, Victor, *Structured Analysis*, New York: Yourdon Press, 1978.

Yourdon, Edward, *Technique of Program Structure and Design*, Englewood Cliff, NJ: Prentice-Hall, Inc., 1975.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5002	2
3.	Curricular Office, Code 37 Department of Computer Technology Naval Postgraduate School Monterey, California 93943-5000	1
4.	Barry A. Frew, Code 54Fw Department of Administrative Sciences Naval Postgraduate School Monterey, California 93943-5000	4
5.	Kenneth J. Euske, Code 54Eu Department of Administrative Sciences Naval Postgraduate School Monterey, California 93943-5000	1
6.	James M. Graham 831 Skylark Drive Columbus, Mississippi 39702	1

INITIAL DISTRIBUTION LIST

[illegible]

Thesis
G65243 Graham
c.1 The completed Management Information System for the Monterey Navy Flying Club.

35559

10 MAY 89

Thesis
G65243 Graham
c.1 The completed Management Information System for the Monterey Navy Flying Club.